ARCHITECT: THE ARCHITECTURE-BASED TECHNOLOGY EVALUATION AND CAPABILITY TRADEOFF METHOD

A Thesis
Presented to
The Academic Faculty

by

Kelly A. Griendling

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology
December 2011

ARCHITECT: THE ARCHITECTURE-BASED TECHNOLOGY EVALUATION AND CAPABILITY TRADEOFF METHOD

Approved by:

Professor Dimitri N. Mavris, Advisor, Committee Chair
School of Aerospace Engineering
Georgia Institute of Technology

Professor Brian German
School of Aerospace Engineering
Georgia Institute of Technology

Professor Daniel Schrage
School of Aerospace Engineering
Georgia Institute of Technology

Ms. Kelly Cooper
Ships Systems and Engineering Division
Office of Naval Research

Dr. Santiago Balestrini
School of Aerospace Engineering
Georgia Institute of Technology

Professor Charles Dickerson
Electronic and Electrical Engineering
Loughborough University

Date Approved: November 11, 2011

To my son,

Alexander Daniel Cooksey

ACKNOWLEDGEMENTS

I would like to show my thanks to the many people who have helped and supported me in numerous ways throughout the process of writing this thesis. Without the support, guidance, and advice of my committee, Dr. Dimitri Mavris, Dr. Daniel Schrage, Dr. Santiago Balestrini, Dr. Brian German, Mrs. Kelly Cooper, and Dr. Charles Dickerson, this thesis would not have been possible. Thank you to all of you for all of the excellent feedback and advice, and for keeping me on track throughout the process. Thank you for taking the time and the effort to guide and advise me, and for not letting me do less than my best. I would like to especially thank my advisor, mentor, and friend, Dr. Mavris. I could not have asked for a more supportive or dedicated advisor, and I am thankful for both all of his advice and the abundance of opportunities with which I have been provided as his student. I would also like to give a special thank you to Mrs. Kelly Cooper and the Office of Naval Research, for believing in this research enough to provide the funding to support it, and for continually giving insightful and stimulating feedback which has continually improved the quality of the research.

I am indebted to my many of my colleagues for their continual encouragement, feedback, and emotional support throughout my years at ASDL. First and foremost, I would like to thank the ARCHITECT Team (Joe Iacobucci, Charles Domercant, Burak Bagdatli, Annie Jones, Shuo-ju Chou, and David Warshawsky), for not only believing in this research enough to join to the team, but for all of your honesty and kindness, and for all of your excellent work on a daily basis. I could not have asked for better teammates and friends, and you make me look forward to coming to work every day. The ARCHITECT project has been successful because of you, and I am

looking forward to continued success in the future years. I have worked with many other great teams during my time here, and I would like to thank the VSE team, the BAE team, the JFCOM team, and the ASDL Architecture Working group for making ASDL a great place to work and learn. I would like to also thank the many friends I have had in the lab over the past few years for your support, encouragement, and friendship. I would like to give a special thank yous to Michael Balchanos, Elizabeth Saltmarsh, Scott Wilson, Megan Halsey, William Engler, and Hernando Jimenez, who have all been amazingly generous with your friendship and support. Finally, I would like to give a very special thank you to John Salmon, for not only being a great friend and colleague, but also for teaching me to JMP. Without you, I really might have never finished.

I owe my deepest gratitude to my husband Daniel, for his love and support, for many nights of taking care of everything at home, and for endless discussions on the technical content of this thesis. Second, I would like to thank my son Alex, whose smile never fails to erase all stress, and who makes every day a happy one. Thank you both for your unwavering patience during all of my late nights and stressful days. I would also like to thank my parents for giving me an interest in science and engineering and for believing in me throughout my life. Thank you for never letting me quit and for giving me excellent role models in every aspect of life. Also, I would like to thank my mom especially for being my editor on this thesis. My brothers and sister have also given me much of their love and support, and for this I am very grateful.

Finally, to all of my friends outside of ASDL, who I have barely seen in the last few years as I was trying to finish, thank you for not giving up on me and dragging me out of my black thesis hole on a regular basis. To Mollye Nardi, thank you being the best study buddy in the world, for making me stay focused, and thank you for your friendship over the past 18 years. To Michael Kingsley, Lauren Wagner, Evan

and Stephen Hopkins, Beth Thompson, Ryan Brown, and Stephen Tongelidis, thank for all the years of love friendship, and support during the Ph.D. process.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

ABM Agent-Based Model

ARC-VM Architecture Real Options Complexity-Based Valuation Methodology

ARCHITECT Architecture-based Technology Evaluation and Capability Tradeoff Method

ARCNET Architecture Resource-based Collaborative Network Evaluation Tool

C2 Command and Control

C4ISR Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance

CADM Core Architecture Data Model

CBA Capabilities-Based Assessment

CCA Clinger Cohen Act

CDM Conceptual Data Model

CER Concept Exploration and Refinement

CONOPS Concept of Operations

CONUS Continental United States

CPM Critical Path Method

CV Capability Viewpoint

CVN Aircraft Carrier, Nuclear Variant

DAPS Defense Acquisition Program Support

DES Discrete Event Simulation

DIV Data and Information Viewpoint

DM2 DoDAF Meta Model

DoD Department of Defense

DoDAF Department of Defense Architecture Framework

DoE Design of Experiments

DOTMLPF Doctrine, Organization, Training, Materiel, Leadership and education, Personnel and Facilities

DSM Design Structure Matrix

FDC Functional Domain Complexity

FOB Forward Operating Base

FOI Swedish National Defence Research Agency

FPC Functional Processing Complexity

G/Q/M Goal/Question/Metric

GAO General Accounting Office

HOQ House of Quality

ICAMS Integrated C4ISR Analysis and Management System

ICD Initial Capabilities Document

IOL Interoperability Level

IRMA Interactive Reconfigurable Matrix of Alternatives

IV Independent Variable

JCIDS Joint Capability Integration and Development System

JFCOM Joint Forces Command

KPP Key Performance Parameter

LDM Logical Data Model

LISI Levels of Information System Interoperability

M&S Modeling and Simulation

MC Markov Chain

MCS Monte Carlo Simulation

MDA Missile Defense Agency

MoA Matrix of Alternatives

MoE Measure of Effectiveness

MoFEs Measures of Force Effectiveness

MoP Measure of Performance

NVAC National Visualization and Analytics Center

O&S Operations and Support

OCONUS Outside of the Continental United States

OV Operational Viewpoint

PAID Procedures, Application, Infrastructure, and Data

PERT Program Evaluation and Review Technique

PES Physical Exchange Scheme

PSC Preferred System Concept

PSM Practical Systems/Software Measurement

PV Project Viewpoint

QFD Quality Function Deployment

RGS Requirements Generation System

ROSETTA Relational-Oriented Systems Engineering for Technology Tradeoff Analysis

RPC Resource Processing Complexity

RSC Resource State Complexity

RSE Response Surface Equation

RSM Response Surface Methodology

SD System Dynamics

SE Systems Engineering

SME Subject Matter Expert

SOF Special Operations Forces

SoS System of Systems

SoSE System of Systems Engineering

SPN Stochastic Petri Net

StdV Standards Viewpoint

SV Systems Viewpoint

SvcV Services Viewpoint

TD Technology Development

TRL Technology Readiness Level

V&V Verification and Validation

SUMMARY

The use of architectures for the design, development, and documentation of system-of-systems engineering has become a common practice in recent years. This practice became mandatory in the defense industry in 2004 when the Department of Defense Architecture Framework (DoDAF) Promulgation Memo mandated that all Department of Defense (DoD) architectures must be DoDAF compliant. Despite this mandate, there has been significant confusion and a lack of consistency in the creation and the use of the architecture products. Products are typically created as static documents used for communication and documentation purposes that are difficult to change and do not support engineering design activities and acquisition decision making. At the same time, acquisition guidance has been recently reformed to move from the bottom-up approach of the Requirements Generation System (RGS) to the top-down approach mandated by the Joint Capabilities Integration and Development System (JCIDS), which requires the use of DoDAF to support acquisition. Defense agencies have had difficulty adjusting to this new policy, and are struggling to determine how to meet new acquisition requirements.

This research has developed the Architecture-based Technology Evaluation and Capability Tradeoff (ARCHITECT) Methodology to respond to these challenges and address concerns raised about the defense acquisition process, particularly the time required to implement parts of the process, the need to evaluate solutions across capability and mission areas, and the need to use a rigorous, traceable, repeatable method that utilizes modeling and simulation to better substantiate early-phase acquisition decisions. The objective is to create a capability-based systems engineering methodology for the early phases of design and acquisition (specifically Pre-Milestone

A activities) which improves agility in defense acquisition by (1) streamlining the development of key elements of JCIDS and DoDAF, (2) moving the creation of DoDAF products forward in the defense acquisition process, and (3) using DoDAF products for more than documentation by integrating them into the problem definition and analysis of alternatives phases and applying executable architecting. This research proposes and demonstrates the plausibility of a prescriptive methodology for developing executable DoDAF products which will explicitly support decision-making in the early phases of JCIDS. A set of criteria by which CBAs should be judged is proposed, and the methodology is developed with these criteria in mind. The methodology integrates existing tools and techniques for systems engineering and system of systems engineering with several new modeling and simulation tools and techniques developed as part of this research to fill gaps noted in prior CBAs.

A suppression of enemy air defenses (SEAD) mission is used to demonstrate the application of ARCHITECT and to show the plausibility of the approach. For the SEAD study, metrics are derived and a gap analysis is performed. The study then identifies and quantitatively compares system and operational architecture alternatives for performing SEAD. A series of down-selections is performed to identify promising architectures, and these promising solutions are subject to further analysis where the impacts of force structure and network structure are examined. While the numerical results of the SEAD study are notional and could not be applied to an actual SEAD CBA, the example served to highlight many of the salient features of the methodology. The SEAD study presented enabled pre-Milestone A tradeoffs to be performed quantitatively across a large number of architectural alternatives in a traceable and repeatable manner. The alternatives considered included variations on operations, systems, organizational responsibilities (through the assignment of systems to tasks), network (or collaboration) structure, interoperability level, and force structure. All of the information used in the study is preserved in the environment, which is dynamic

and allows for on-the-fly analysis. The assumptions used were consistent, which was assured through the use of single file documenting all inputs, which was shared across all models. Furthermore, a model was made of the ARCHITECT methodology itself, and was used to demonstrate that even if the steps took twice as long to perform as they did in the case of the SEAD example, the methodology still provides the ability to conduct CBA analyses in less time than prior CBAs to date. Overall, it is shown that the ARCHITECT methodology results in an improvement over current CBAs in the criteria developed here.

CHAPTER I

INTRODUCTION AND MOTIVATION

The practice of applying architectures to the design, development, and documentation of system-of-systems engineering has become common in recent years, and became mandatory in the defense industry after the Department of Defense (DoD) issued the Department of Defense Architecture Framework (DoDAF) Promulgation Memo in 2004, mandating that all DoD architectures must be DoDAF compliant. Following this mandate, significant confusion and a lack of consistency in the creation and the use of the architecture products has persisted. Often, architecture descriptions are created as static documents used only for communication and documentation purposes. These static descriptions are difficult to update and do not support engineering activities and acquisition decision making. In 2002, acquisition guidance was reformed, moving from the bottom-up Requirements Generation System (RGS) to a top-down approach taken by the new the Joint Capabilities Integration and Development System (JCIDS). JCIDS also required the use of some DoDAF architecture products to support various phases of the acquisition process. However, defense agencies have had difficulty adjusting to this new policy, and are struggling to determine how to meet new acquisition requirements. In fact, studies conducted under the new system have undertaken much criticism, as has the new process itself. This research attempts to respond to these challenges and address concerns raised about the defense acquisition process by developing and demonstrating the plausibility of a new methodology, which is named the Architecture-based Technology Evaluation and Capability Tradeoff (ARCHITECT) method.

Due to the relatively new nature and constant fluctuation in some of the relevant

fields for this research, this chapter will begin by presenting definitions of some of the key terminology, and brief discussion of the relevant defense acquisition policies and standards. It will then highlight some of the challenges that have arisen during the transition to these new policies and standards, and conclude with an overall research objective. The second chapter begins by presenting perspectives on similar acquisition problems in other fields, discussing the existing state-of-the-art in defense acquisition guidance, identifying the general steps required for the ARCHITECT method, and developing the Research Questions on a step-by-step basis that guide the remainder of the thesis. The third chapter will provide background material on the key enablers and alternative techniques for each step of the methodology. These are used to formally develop the methodology in the fourth chapter, which also develops the criteria for judging the success of the methodology at improving CBA. As the method is developed, these criteria are used to determine which of the enablers discussed in the background section is most appropriate for the ARCHITECT method. The methodology is then demonstrated using a suppression of enemy air defenses (SEAD) example scenarios in the fifth chapter. Finally, the conclusions chapter summarizes the findings and contributions of this thesis, and suggests the next steps in this research area.

1.1 Terminology

1.1.1 Complex Systems

While there are a variety of definitions of complex systems available in literature, these definitions share a set of common themes. Jackson [94] has complied a set of characteristics of complex systems that has been summarized by [106] to include:

- Large number of variables or elements
- Rich interactions among elements
- Difficulty in identifying attributes and emergent properties

- Loosely organized (structured) interaction among elements
- Probabilistic, as opposed to deterministic, behavior in the system
- System evolution and emergence over time
- Purposeful pursuit of multiple goals by system entities or subsystems (pluralistic)
- Possibility of behavioral influence or intervention in the system
- Largely open to the transport of energy, information, or resources from/to across the system boundary to the environment
- Diverse in technology, context, operation, geography, and conceptual frame

1.1.2 Systems of Systems

In order to define a system of systems (SoS), it is first necessary to have a clear definition of a system. A system is defined in this research to be a "functionally, physically, and/or behaviorally related group of regularly interacting or interdependent elements; that group of elements forming a unified whole" [100]. The DoD defines an SoS as "set or arrangement of systems that results when independent and useful systems are integrated into a larger system that delivers unique capabilities" [40] or, alternately, "Groups of systems, each of which individually provides its own mission capability, that can be operated collectively to achieve an independent, and usually larger, common mission capability" [6]. In this context, independence is often taken to imply different stakeholders. Upon first glance, the definition for a system and SoS appear to be very similar. Both describe a group of elements which are somehow integrated into a whole to accomplish a given behavior. Thus, further clarification is needed to fully distinguish between what is meant by a system and SoS in this research. Looking at the definition of SoS from a variety of sources both within and outside

of the defense community gives a wide variety of definitions, implying that the term is not precisely defined. However, by compiling these definitions, it can be said that compared to a system, an SoS might[115, 19, 57, 89, 153]:

- be larger in scope
- have more complex integration
- be subject to higher degree of uncertainty and risk
- evolve more continuously with elements of differing lifecycles
- lack a single management/acquisition entity and have a broader range of stakeholders
- have elements that are not designed to fit the whole, and that are integrated post-design and deployment
- exhibit emergent behaviors
- have more ambiguous requirements and fuzzy boundaries
- have geographic distribution
- have elements with operational independence
- have continuous systems engineering that is never finished

Furthermore, it can be observed that SoS often include the themes of [115, 19, 57, 89, 153]:

- autonomy
- collaboration
- complexity

- heterogeneity
- adaptability
- emergent behavior
- dependability
- being distributed.

The commonalities between the definition of complex system and SoS implies that while a complex system is not necessarily an SoS, an SoS is almost always a complex system.

As can be seen from the above definitions, the design of an SoS presents a hierarchical series of design challenges. The overall architecture must be designed to maximize the overall delivered capability, and the individual systems must be designed to adequately perform their role in the architecture. This kind of thinking means that systems should not necessarily be optimized based on their own individual performance, but rather on their performance in the operational context. The important thing to note about a system is that it generally can operate independently of the architecture to perform some useful function. However, when many systems are integrated into the architecture to form a working SoS, there is some behavior that is greater than the sum of the individual pieces.

Types of System of Systems

SoS have been classified into four general categories, *virtual* SoS, *acknowledged* SoS, *collaborative* SoS, and *directed* SoS.

A virtual SoS is defined as follows by the DoD [40]:

> Virtual SoS lack a central management authority and a centrally agreed upon purpose for the SoS. Large-scale behavior emerges—and may be

desirable—but this type of SoS should rely upon relatively invisible mechanisms to maintain it.

A collaborative SoS is defined as follows by the DoD [40]:

> In collaborative SoS, the component systems interact more or less voluntarily to fulfill agreed upon central purposes. The Internet is a collaborative system. The Internet Engineering Task Force works out standards but has no power to enforce them. The central players collectively decide how to provide or deny service, thereby providing some means of enforcing and maintaining standards.

An acknowledged SoS is defined as follows by the DoD [40]:

> Acknowledged SoS have recognized objectives, a designated manager, and resources for the SoS; however, the constituent systems retain their independent ownership, objectives, funding, and development and sustainment approaches. Changes in the systems are based on collaboration between the SoS and the system.

A directed SoS is defined as follows by the DoD [40]:

> Directed SoS are those in which the integrated SoS is built and managed to fulfill specific purposes. It is centrally managed during long-term operation to continue to fulfill those purposes as well as any new ones the system owners might wish to address. The component systems maintain an ability to operate independently, but their normal operational mode is subordinated to the central managed purpose.

Although there are examples of all four types of systems within the DoD, the most prevalent type of SoS seen in the DoD is the acknowledged SoS. The Missile Defense Agency's Ballistic Missile Defense System is an example of an acknowledged SoS.

The Defense Acquisition Guidebook provides the following examples for the other types of SoS: "The Future Combat System is the best-known example of a 'directed SoS.' Communities of interest are good examples of DoD 'collaborative SoS,' and the Global Information Grid is the predominant DoD 'virtual SoS"' [40].

1.1.3 Systems Engineering/Architecting

The practice of architecting is defined by IEEE and ISO [93] to be the "process of conceiving, defining, expressing, documenting, communicating, certifying proper implementation of, maintaining and improving an architecture throughout a system's life cycle". This is similar to the definition of systems engineering provided by IEEE, which is "an interdisciplinary approach to derive, evolve, and verify a life-cycle balanced system solution that satisfies customer expectations and meets public acceptability" [88]. INCOSE uses a similar definition, saying that "Systems Engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, and then proceeding with design synthesis and system validation while considering the complete problem. Systems Engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs" [89]. Thus it may be said that the process of architecting is ultimately the application of system engineering to the architecture of the system.

1.1.4 System of Systems Engineering

SoS Engineering (SoSE) is, according to the SoSE Center of Excellence under OSD, "an emerging interdisciplinary approach focusing on the effort required to transform capabilities into SoS solutions and shape the requirements for systems. SoS Engineering ensures that: Individually developed, managed, and operated systems function as

autonomous constituents of one or more SoS and provide appropriate functional capabilities to each of those SoS, Political, financial, legal, technical, social, operational, and organizational factors, including the stakeholders' perspectives and relationships, are considered in SoS development, management, and operations, an SoS can accommodate changes to its conceptual, functional, physical, and temporal boundaries without negative impacts on its management and operations, an SoS collective behavior, and its dynamic interactions with its environment to adapt and respond, enables the SoS to meet or exceed the required capability" [159]. SoSE is essentially the practice of SE applied to an SoS in such a way as to address the unique challenges of an SoS.

1.1.5 Capability

Capability has multiple definitions within the DoD. The DoD Dictionary defines a capability to be "the ability to execute a specified course of action (A capability may or may not be accompanied by an intention.)" [102]. This definition is less descriptive than the definition found within acquisition guidance, which defines a capability to be "The ability to achieve a desired effect under specified standards and conditions through combinations of means and ways across the doctrine, organization, training, materiel, leadership and education, personnel, and facilities (DOTMLPF) to perform a set of tasks to execute a specified course of action" [31]. For this research, the latter definition will be used.

Although this definition provides a starting point for defining a capability, there is still ambiguity. Examining and comparing examples of capabilities provided by a range of organizations exemplifies this ambiguity.

1.1.6 Architecture

An SoS is considered to have an underlying architecture, which is defined by ANSI/ IEEE 1471-2000 [10] to be "The fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the

principles governing its design and evolution." The ISO/IEC/IEEE 42010 standard uses a similar definition, with the addition of the environment, defining architecture as the "fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution" [93]. However, the word architecture did not originate in the field of engineering. Historically, architecture has referred to the art and practice of designing and constructing buildings [138]. The WordNet English lexical database developed by Princeton University defines architecture as "the discipline dealing with the principles of design and construction and ornamentation of fine buildings; architecture and eloquence are mixed arts whose end is sometimes beauty and sometimes use" [146]. There have been many definitions of architecture in this context, but they share three coherent themes. An architecture has a structure, a utility (or intention), and must be beautiful or attractive [54]. These could be considered the three pillars of successful architecture. Engineering definitions, like the one stated above, capture the structural component of architecture, but leave out the concepts of utility and attractiveness. These concepts, however, should not be disregarded when applying the principles of architecture to engineering. Everything that an engineer designs has been created with a purpose, in order to do something, or fulfill some need. This applies to SoS as well. An SoS is created with the intention of performing a specific mission, which is the utility of the SoS. In addition, an engineer's creation must be attractive to stakeholders, including the buyer and user, among others. While this beauty may be slightly different in nature than the beauty of traditional building architecture, it is nonetheless important to having a product accepted and used. In the context of SoS Engineering (SoSE), attractiveness may include things such as the affordability, the ease of use, the reliability, etc. In light of these observations, the definition of architecture will be modified for this research. Here, architecture will be defined as

"The fundamental organization of a system, embodied in its components, their relationships to each other and the environment, the principles governing its design and evolution, its purpose, and its attractiveness (e.g. functionality, cost, etc)."

Furthermore, it should be noted that the word architecture comes from the Greek word arkhitektonike, which is a combination of two words meaning 'chief' and 'builder' [62]. Thus the architect is the chief builder, meaning that the architect is responsible for overseeing all aspects of building, and is essentially the integrator of all aspects of building design. The same is true of an architect in the context of engineering. The architect is responsible for bridging the communication gap between stakeholders and engineers, and translating stakeholder desires into actionable engineering requirements.

1.1.7 Framework

DODAF defines an architecture framework as "providing rules and guidance for structuring, classifying, and organizing architectures" [45]. It further defines an architecture description as a "representation of a defined domain, as of a current or future point in time, in terms of its component parts, how those parts function, the rules and constraints under which those parts function, and how those parts relate to each other and the environment" [45].

1.1.8 Model

When discussing architectures, it is important to understand the relationship between the representations of architectures within frameworks and models. A model will be defined here using the definition provided by D. T. Ross, which is:

"M is a model of A with respect to question set Q if and only if M may be used to answer questions about A in Q within tolerance T" [150].

This definition implies that an architecture represented in a framework is a model of an SoS if it can be used to answer questions to a specified tolerance. The outputs

of representing a particular system or SoS in an architecture framework are typically called artifacts, products, or views. In this research, these terms will be used interchangeably, although there is some debate in the community as to which terms are most appropriate. For example, DoDAF has changed its terminology from "products" in version 1.5 to "models" in version 2.0 in order to better reflect the emphasis on data-centricity [50]. However, in general, the views or products that are created as the result of depicting something in an architecture framework, can be considered models by the above definition if and only if they are able to be used to help answer questions about the system or SoS within an acceptable tolerance. Thus, in order to determine if a view is an appropriate model for a given situation, it must be known what questions a user would like to answer and what the required tolerances are for that answer. An architecture view can only be considered a model when it is able to satisfactorily answer these questions. In some cases, something as simple as a pictorial representation of the system or system-of-systems may be sufficient. However, in many cases, an executable simulation will need to be associated with the view in order for that view to be an appropriate model.

1.2 Department of Defense Acquisition

1.2.1 Defense Acquisition Overview

In the defense industry, acquisition and early-phase engineering guidance have been completely re-written in the last decade in an attempt to tackle problems stemming from the previous stove-piped Requirements Generation System (RGS) [31]. In fact, in 2002, then Secretary of Defense Donald Rumsfeld wrote of the RGS, "It is pretty clear it is broken, and it is so powerful and inexorable that it invariably continues to require things that ought not to be required, and does not require things that need to be required" [152]. This memo sparked the creation of the Joint Capabilities Integration and Development System (JCIDS), which has replaced the RGS since 2003

in supporting requirements development portion of capability portfolio management for the Department of Defense (DoD). JCIDS was created with three primary principles in mind. First, needs should be described in terms of required capabilities, rather than specific system requirements. This shift in thinking was due to a recognition that system requirements often did not necessarily map to improving specific operational objectives. Thus, requirements should flow down from operational requirements that provide specific goals of actions that need to be enabled. The second guiding principle of JCIDS is that needs should be derived from a joint perspective and joint set of concepts that not only enumerate the best way to operate with the existing resources, but also provide room for future improvements across the entire spectrum of military operations. The third principle is that a single general or flag officer should oversee each functional portfolio so that efforts were less stove-piped and there was one central point of contact responsible for knowing what was going on across entire domains (e.g. command and control) [101].

The current Defense Acquisition policy utilizes three interacting systems for support of acquisition decision making. These are generally referred to as the DoD Decision Support Systems (DSS). The Planning, Programming, Budgeting, and Execution (PPB&E) Process is used for strategic planning and portfolio management. It helps to determine how resources will be allocated and which plans and programs are most appropriate to satisfy national security needs. JCIDS is geared toward understanding, identifying, and prioritizing capability gaps and proposing solutions to fill those gaps. It is used to manage requirements and verify that proposed solutions effectively meet requirements in a timely and affordable manner. Finally, the Defense Acquisition System (DAS) oversees the acquisition of weapons and automated information systems [40]. All of these DSS are largely focused on the acquisition of materiel solutions. However, in order to start a materiel acquisition program, it must first be determined that a materiel acquisition is needed to fill capability gaps, and

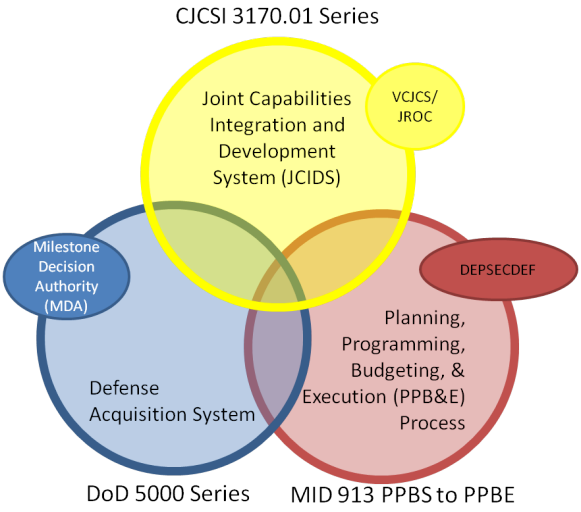


Figure 1: DoD Decision Support Systems (adapted from [40])

that those gaps could not be adequately filled using another type of solution. This is done using a Capabilities-Based Assessment (CBA) that is performed before the first JCIDS Milestone (called Pre-Milestone A) and is a prerequisite to entering the JCIDS process. The CBA is used to identify and prioritize capability gaps and to explore both materiel and non-materiel solutions to these gaps. The full solution space is often referred to as doctrine, organization, training, materiel, leadership and education, personnel and facilities (DOTMLPF) solutions. Non-materiel solutions are considered to be those other than materiel, and are often referred to as DOT_LPF solutions [31]. Figure 1 shows the three DoD DSS with the overseeing organization and official guiding documents overlaid.

1.2.2 Department of Defense Directive 5000

The DoD 5000 provides management principles and mandatory policies regarding the management of all DoD Acquisitions. The DoD 5000 is comprised of two documents, the DoD 5000.01 and the DoD 5000.02. The DoD 5000.01 is a directive that provides the overall policy and management guidance, while the DoD 5000.02 in an instruction that formalizes and elaborates on the management framework for acquisition [47,

49]. The following excerpt from the DoD 5000.01 provides a summary of the policy objectives of the DoD 5000, and has been copied directly from the DoD 5000.01 [47]:

> *POLICY*
>
> *4.1. The Defense Acquisition System exists to manage the nation's investments in technologies, programs, and product support necessary to achieve the National Security Strategy and support the United States Armed Forces. The investment strategy of the Department of Defense shall be postured to support not only today's force, but also the next force, and future forces beyond that.*
>
> *4.2. The primary objective of Defense acquisition is to acquire quality products that satisfy user needs with measurable improvements to mission capability and operational support, in a timely manner, and at a fair and reasonable price.*
>
> *4.3. The following policies shall govern the Defense Acquisition System:*
>
> *4.3.1. Flexibility. There is no one best way to structure an acquisition program to accomplish the objective of the Defense Acquisition System. MDAs and PMs shall tailor program strategies and oversight, including documentation of program information, acquisition phases, the timing and scope of decision reviews, and decision levels, to fit the particular conditions of that program, consistent with applicable laws and regulations and the time-sensitivity of the capability need.*
>
> *4.3.2. Responsiveness. Advanced technology shall be integrated into producible systems and deployed in the shortest time practicable. Approved, time-phased capability needs matched with available technology and resources enable evolutionary acquisition strategies. Evolutionary acquisition strategies are the preferred approach to satisfying operational needs. Spiral development is the preferred process for executing such strategies.*

4.3.3. Innovation. Throughout the Department of Defense, acquisition professionals shall continuously develop and implement initiatives to streamline and improve the Defense Acquisition System. MDAs and PMs shall examine and, as appropriate, adopt innovative practices (including best commercial practices and electronic business solutions) that reduce cycle time and cost, and encourage teamwork.

4.3.4. Discipline. PMs shall manage programs consistent with statute and the regulatory requirements specified in this Directive and in reference (b). Every PM shall establish program goals for the minimum number of cost, schedule, and performance parameters that describe the program over its life cycle. Approved program baseline parameters shall serve as control objectives. PMs shall identify deviations from approved acquisition program baseline parameters and exit criteria.

4.3.5. Streamlined and Effective Management. Responsibility for the acquisition of systems shall be decentralized to the maximum extent practicable. The MDA shall provide a single individual with sufficient authority to accomplish MDA-approved program objectives for development, production, and sustainment. The MDA shall ensure accountability and maximize credibility in cost, schedule, and performance reporting.

In particular, this research focuses on addressing the call for flexibility, responsiveness, innovation, and discipline as key components to successful acquisition. A successful methodology to aid in any phase of the acquisition process must conform to these principles.

The Defense Acquisition Management System is summarized in Figure 2.

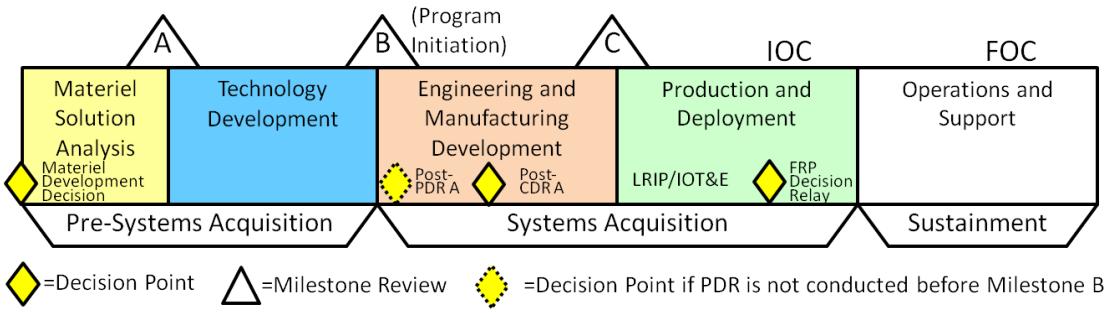


Figure 2: Defense Acquisition Management System (recreated from [49])

1.2.3 Capabilities-Based Assessment and JCIDS

One of the primary components of the JCIDS process is the execution of a CBA. The CBA focuses on determining potential solutions across the doctrine, organization, training, materiel, leadership and education, personnel, or facilities (DOTMLPF) spectrum to fill capability gaps. A CBA identifies the mission to be studied, the capabilities required to perform that mission, the operational characteristics and attributes of each capability, existing capability gaps and operational risks, an assessment of the viability of non-materiel solutions, and, if needed, a recommendation on the type of materiel solution to be pursued. A CBA also justifies that a solution is needed for the identified gaps, as opposed to accepting the operational risk and making no changes. A materiel solution can fall into one of three categories: transformational, evolutionary, or information technology. The CBA results in two documents, depending on the approach recommended. If a non-materiel solution is recommended, a DOTMLPF Change Recommendation (DCR) is created instead. If a materiel solution is indicated, an Initial Capabilities Document (ICD) is presented to the Joint Requirements Oversight Committee (JROC) for review. Once the ICD is approved, it is sent to the Milestone Decision Authority (MDA) to determine the scope of the analysis of alternatives (AoA) that follows, as well as designating the lead components in the Materiel Development Decision (MDD). This leads to a materiel solutions analysis phase, which, in conjunction with the ICD, supports entrance into

JCIDS Milestone A [31]. Because the CBA is the kickoff to the JCIDS process, doing a poor job on a CBA is likely to lead to solutions that are not affordable, untimely, or ineffective [101].

The JCIDS process itself has two further milestones. Milestone B is preceded by a Technology Development Phase, which results in the creation of a Capabilities Description Document (CDD). This includes the design, development, testing, and prototyping of the solution through the Preliminary Design Review (PDR). The CDD specifies the performance attributes of the system designed to fill capability gaps. Key Performance Parameters (KPPs) are identified and validated, and the risks associated with meeting the KPPs are assessed. JROC must approve the CDD, after which the MDA makes the decision as to whether or not to pursue development at Milestone B. If development is pursued, the Engineering Manufacturing and Development phase is initiated, at the end of which a Capabilities Production Document (CPD) is produced. The CPD documents the actual performance of the system, and compares it to KPP thresholds specified in the CDD. The JROC will then assess whether the performance of the system is adequate to give the operational advantages needed, and if so, will approve the CPD. The MDA will then decide to enter in system production and deployment. This decision occurs at Milestone C [31].

1.2.4 Challenges in Defense Acquisition

Despite the recent acquisition reform, the DoD is still facing significant challenges in weapon system acquisition. A 2008 GAO report states that, "DoD has taken some action to improve acquisition outcomes, but its weapons programs continue to take longer, cost more, and deliver fewer capabilities than originally planned. These persistent problems, coupled with current operational demands, have impelled DoD to work outside of its traditional acquisition process to acquire equipment that meet urgent

warfighter needs" [174]. Another 2008 GAO Report, titled "DoD's Requirements Determination Process Has Not Been Effective in Prioritizing Joint Capabilities", states that "proposals for new capability needs and system solutions are not systematically prioritized across capability and mission areas, and virtually all proposals that have gone through JCIDS have been validated. The JCIDS process has also proven to be lengthy, taking on average up to 10 months to validate a need. Such a protracted process further undermines the department's efforts to effectively respond to the needs of the warfighter, especially those that are near term" [173]. JCIDS itself has undertaken severe criticism from many entities, including a report by the Defense Business Board which described JCIDS as "lots of process for process' sake" [41]. It has also been described as "new, ambitious, evolving, and far from perfect" [96].

CBA has also undertaken much criticism. Captain(R) Norbert Doerry stated that "The problem with this process is that each mission area has its own CBA, yet ships and aircraft are inherently multi-mission" [58]. He further goes on to state that "the current process does not facilitate cost-performance trade-offs at the fleet and force architecture level" [58]. It has also been observed that the definition of what a CBA is and what should be included is ambiguous [96]. Furthermore, CBAs have been criticized for being very time consuming. In fact, a 2006 white paper stated that "None of the JROC-commissioned CBAs done to date have been able to finish in less than a year" [96]. In fact, some organizations have issued additional CBA guidance in order to "expedite and standardize" the CBA process [163].

In the 2009 GAO Report on Defense Acquisitions, it was reported that [175]:

> Since fiscal year 2000, the Department of Defense (DoD) has significantly increased the number of major defense acquisition programs and its overall investment in them. However, acquisition outcomes have not improved. In most cases, the programs we assessed failed to deliver capabilities when

promised–often forcing warfighters to spend additional funds on maintaining legacy systems.

While the report acknowledges that recent changes to the acquisition system could help mitigate some of these challenges, they state that additional actions will be required by the DoD in order to improve the outcome of weapons acquisition programs. The five specific actions suggested are [175]:

1. Making better decisions about which programs should be pursued or not pursued given existing and expected funding;
2. Developing an analytical approach to better prioritize capability needs;
3. Requiring new programs to have manageable development cycles;
4. Requiring programs to establish knowledge-based cost and schedule estimates; and
5. Requiring contractors to perform detailed systems engineering analysis before proceeding to system development.

Of these five recommendations, the first four should be completely or partially addressed during the CBA stage of acquisition. The Weapon System Acquisition Reform Act of 2009 was implemented to address some of the challenges, but was focused on improving the efficiency of major defense acquisition programs once they have been established, but did little to reform the CBA process by which the initial decision is made as to whether an acquisition program should be begun in the first place [1].

1.3 Engineering Architectures

1.3.1 Rationale for an Architecture Based Approach

The use of architectures for engineering did not originate in the defense or systems engineering sectors. The concept was originally applied in the field of software engineering. The first to recognize the role of structure in programming was Edsger Dijkstra in 1968. Dijkstra asserted that to obtain a high confidence level for correctness in the case of really large computer programs, individual components must all have a very high probability of correctness. This is based on the simple probabilistic relationship P=pN, where P is the probability of correctness of the entire program, p is the probability of correctness of individual model, and N is the number of modules. Since in a large program, N is very large, p must be extremely close to 1 in order for P to significantly greater than 0. His claim was that since it is the programmer's responsibility not only to produce a correct program, but also to be able to demonstrate the correctness in a convincing way, the program must be well structured. This structure must not only be useful in demonstrating correctness, but also in creating program adaptability and manageability. He goes on to claim that programming should be considered as the minimization of a cost/performance ratio, and that "the art of programming is the art of organizing complexity, of mastering multitude and avoiding its bastard chaos as effectively as possible" [56]. These ideas led him to develop the concept of structured programming, which was the basis for the future ideas of software architecting.

In 1972, David L. Parnas published a paper titled "On the Criteria to be Used in Decomposing Systems into Module", which made the claim that modularization is "a mechanism for improving the flexibility and comprehensibility of a system while allowing the shortening of its development time" [142]. Parnas' based his modularization philosophy on the idea of isolating difficult or high-risk (i.e. likely to change) design decisions. Thus, when there was a change in requirements or design of one

module, the amount of rework would be minimized [142]. Both Parnas and Dijkstra emphasized not only the concept of developing a structure at the onset of a software development activity, but also the importance of the structure in the timeliness of the effort as well as the correctness of the resulting product. These ideas formed the basis for what would become known as "software architecting".

The discipline of software architecting took off in the 1990's. Much research was done on types of architectures, formal descriptive languages for architectures, and formal methods for architecting. In 1993, David Garlan and Mary Shaw published "An Introduction to Software Architecture" [73], which provided a survey of existing architectural styles and outstanding problems in the field and presented several case studies. It was not until 2000, however, that the first formal standard for software architecting was published. This standard is ANSI/IEEE 1471-2000: Recommended Practice for Architecture Description of Software-Intensive Systems. This standard asserts that using architectures for software development gives the benefits of "reduced costs and increased quality, such as usability, flexibility, reliability, interoperability and other system qualities" [10].

The Department of Defense Architecture Framework (DODAF) states two primary reasons for using architectures in defense. The first is that it is mandated by law through the Clinger-Cohen Act (enacted in 1996) [168] and Office of Management and Budget (OMBa130) Circular A-130 (published in 2000) [132]. The second is that "experience has demonstrated that the management of large organizations employing sophisticated systems and technologies in pursuit of joint missions demands a structured, repeatable method for evaluating investments and investment alternatives, as well as the ability to effectively implement organizational change, create new systems, and deploy new technologies" [45]. According to Bachmann, et. al., "the initial stages of architecture design are where the most fundamental design decisions are made; these are the decisions that are the most difficult to correct when they are in

error. In order to effectively design a product line architecture, the architect needs a disciplined design method that focuses the creative process; provides a strategy for coping with the uncertainty in requirements; provides guidance in organizing the decisions made during the design process; and makes clear why the steps of the method exist" [12].

Recently, capability based analysis, design, and acquisition has shown a significant presence within defense related programs. The paradigm shift to capabilities-based acquisition within the DoD is causing a fundamental shift in the way defense-related systems are both engineered and purchased. New mission needs and technological advancements have led to new directives that are pushing defense acquisition towards a capability-based approach. In particular, advancements in communication and transportation combined with new and diverse enemies has led to a call for more joint operations, more integrated operations, and a better way of designing and acquiring systems and systems-of-systems to support these needs.

The capability-based mentality shares a natural link with architecting, in that capabilities are achieved through a series of activities. These activities can be represented as an operational architecture. Through the architecting process, these activities can be mapped to candidate solutions, which can then be evaluated and compared. These solutions provide the "ways and means" by which a capability is achieved. This kind of approach has been suggested to help address high level capability needs and help to avoid the "stove-piping" that has often plagued defense acquisition [31].

Acquisition decision-making has traditionally been done using a technology-based approach. That is, a technology is selected based on its ability to meet a specified set of technical requirements. These requirements are often very specific in nature, and are focused on technology maturation rather than capability enhancement. The DoD has recently been shifting towards a capability-based design approach. This approach

focuses less on the technical specifications of a solution, and more on the capabilities provided by the proposed system or systems. For example, instead of a requirement that says "The aircraft needs to have a max level speed of 550 knots," the requirements may instead read "The Air Force must be able to find and destroy time-critical targets." These more general capability-based requirements can then be decomposed into a more specific set of requirements using a hierarchical decomposition. This allows more flexibility in the design process.

There are both advantages and challenges to this approach. The former requirement is very concrete. It is easy to directly measure if this requirement is being met by a suggested solution. However, it does not allow for a design that has a max level speed of 540 knots, but has double the range and payload over competing concepts that achieve the speed requirement. Furthermore it does not allow for a solution that is not an aircraft. The second requirement, however, leaves room for an aircraft concept that has a very low max level speed, but can carry and deploy several high-speed, long range air-to-ground missiles from a great distance, or, for example, an aircraft concept that can travel quickly to the combat zone and drop bombs on the target. It would even allow for a solution such as the use of long-range ballistic missiles to be considered in the design space. As a result, more design freedom is given to engineers, which can result in more creative solutions that better fulfill the actual needs of the customer. One of the primary challenges of using a capability-based approach becomes immediately obvious from this example. It is very easy to compare competing systems based on specified and measurable metrics, such as speed. It is much more difficult to compare systems based on more nebulous capability need statements. A top-down method to evaluate competing solutions based on how well they are predicted to meet capability needs is required, and as part of the aforementioned method, some kind of scale on which capabilities can be measured in a requirement. This method should maintain clear traceability in decision making

and show a clear link between design and acquisition decisions and customer needs.

Many experts in the field believe that using an architecture-based approach can help provide this traceability and provide a structured means for comparing competing concepts and supporting decision making. According to Philipp Charles, chief engineer of SPAWAR Systems Center Charleston, "To effectively acquire complex systems of systems in a capability-based acquisition environment requires that we increase the use of integrated architectures to identify inter-relationships and resolve issues with system integration and interoperability that impact the operational effectiveness of warriors; platforms; command and control; networks; and weapons" [32]. He further states that, "The ultimate goal for capabilities-based architecting is to provide a cost-effective analysis of alternative capabilities, system configurations and options characteristics (schedule, performance and costs) at any level of detail desired by a decision maker, structured so that all analysis and current issues are traceable" [32]. In their book, *Using Architectures for Research Development and Acquisition*, Dickerson et al. state that, "Degradation in combat effectiveness can be caused bypoor or non-existent integration or interoperability. Because integration and interoperability are so critical to combat effectiveness, the entire Family of Systems must be considered in the engineering and acquisition process if decision makers are to choose the most operationally sound, technically feasible, and effective program investments" [53].

Furthermore, it has been recognized by experts in the field that in an SoS context, the architecture often plays a much more critical role to success in meeting requirements than it does in a system context. As Dagli and Kilicay-Ergin state, "In the SoS environment, architecture has more influence on requirements than it does in an environment dominated by a stand-alone complex system" [37]. Despite this need for an architecture-based approach, the challenges of SoS mean that current system architecting methods are not sufficient and need to be expanded upon for SoS. Chen

and Clothier state that "Without a disciplined architecture practice, system architecting has proved to be difficult and frustrating in an SoS evolution context since it cannot be carried out successfully without addressing other related architecture issues" [33]. Several key research areas have been defined, including:

- Assurance of trustworthiness in the SoS [37]
- Assurance of interoperability [37]
- Assurance of large scale design and distributed testing [37]
- Assurance of evolutionary growth [37, 33]
- Dealing with hidden interdependencies [37]
- Guarding against hidden cascading failures [37]
- Dealing with complexity [37]
- Linkage with standard SE processes [33]
- Cross-project and multi-organization architecting [33]

1.3.2 Department of Defense Architecture Framework (DoDAF)

The Department of Defense Architecture Framework (DoDAF) was created to aid in design and decision-making for large and complex systems and systems of systems in a capability-based context. DoDAF version 1.5 was mandated for use at the start of this research, but has since been replaced by DoDAF 2.0 [52]. Because of this, this research uses concepts from both 1.5 and 2.0. Although many concepts are common between the two standards, there are some important differences that will be highlighted. A brief overview of both versions and important concepts will be given here. For further details on either version, refer to Appendix A.

DoDAF was created to aid in design and decision-making for large and complex systems and SoS in a capability-based context. More specifically, DoDAF was created because the DoD recognized that "the management of large organizations employing sophisticated systems and technologies in pursuit of joint missions demands a structured, repeatable method for evaluating investments and investment alternatives, as well as the ability to effectively implement organizational change, create new systems, and deploy new technologies" [45]. DoDAF has evolved from what was originally the C4ISR Architecture Framework, which was developed in response to the Clinger Cohen Act of 1996 [168] and the OMB Circular A-130 [132]. Version 1 was released on June 7, 1996. The C4ISR Architecture Framework aimed to address issues identified in a 1995 Deputy Secretary of Defense directive, which identified the need to make an effort across the DoD to integrate and streamline C4ISR capabilities. Version 1 was followed a year and a half later, on December 18, 1997, by version 2 of the C4ISR Framework. Version 2 was the result of continued development and implementation of early lessons learned, and became mandatory in February of 1998. On August 30, 2003, DoDAF version 1.0 was released, restructuring and expanding the C4ISR Architecture Framework to apply across all general mission areas, not just C4ISR. DoDAF version 1.0 also expanded the C4ISR framework to include information about architecture practices, such as usage, decision support, and analytical techniques. In addition to two documents specifying the standard, DoDAF 1.0 also included a deskbook, aimed at providing guidance to users. DoDAF version 1.0 introduced the Core Architecture Data Model (CADM), which attempted to increase the focus on the data elements required to create architecture products [43]. The DoDAF Promulgation Memo, released February 9, 2004, required that all DoD Architectures approved after December 1, 2003 must be DODAF compliant [44].

DoDAF version 1.5, was released on April 23, 2007. Version 1.5 focuses on enabling the transition of the DoD to net-centric warfare (NCW), which focuses on linkages

and information flows between warfighting elements as a way of increasing combat effectiveness. The addition of net-centric guidance in version 1.5 was accompanied by a further emphasis on the data underlying architecture products rather than the products themselves. Version 1.5 included an improved version of the CADM, in an attempt to streamline the creation and reuse of architecture data. DoDAF version 1.5 includes three volumes. Volume I contains introductory material and information concerning management of the architecture data, Volume II specifies the development of architecture data and products, and Volume III describes the CADM. In addition, an online journal replaced the deskbook as the primary forum for usage guidance [45]. DoDAF version 1.5 specified four "views", the All-View (AV), the Operational View (OV), the Systems and Services View (SV) and the Technical Standards View (TV). Each of these views was supported by a series of specified products.

Despite attempts to move to a more data-centric approach to architecting, DoDAF version 1.5 was criticized for a lack of focus on data and an over-emphasis on products [50]. This, along with other criticisms and recognized weaknesses of DoDAF version 1.5, led to the development of DoDAF version 2.0, which became mandatory on May 28, 2009 through another promulgation memo [52]. This memo mandates that all DoD architectures must meet DoDAF version 2.0 conformance criteria, which include: "The data in a described architecture is defined according to the DM2 concepts, associations, and attributes" and, "The architectural data is capable of transfer in accordance with the PES" [50]. DoDAF version 2.0 has two specific goals. First, it attempts to provide a framework and appropriate guidance that allows the user to develop the architecture content needed to fit the required purpose. Secondly, a more rigorous data model is used with the goal of increasing the "utility and effectiveness" of the architectures produced using the framework [50]. DoDAF version 2.0 does this by replacing the CADM with the DoDAF Meta-Model (DM2), which has three levels, the conceptual data model (CDM), the logical data model (LDM), and the physical

exchange scheme (PES). Each level is aimed at capturing data at a different level of detail [50]. In this way, the architecture data can be used to generate information and views at the level required for a given audience, user, or model. The inclusion of the DM2 in DoDAF version 2.0 is the most significant change with respect to this work, as it rectifies many of the challenges inherent in DoDAF version 1.5 for using DoDAF for executable architecting.

In addition to the creation of the DM2, DoDAF version 2.0 also changed the terminology of "views" and "products" to "viewpoints" and "models" (also sometimes called "views" or "artifacts"). There are eight viewpoints in DoDAF 2.0. The All-Viewpoint (AV) and the Operational Viewpoint (OV) are similar to their DoDAF version 1.5 counterparts. The Systems and Services View was broken apart into two viewpoints, the Systems Viewpoint (SV) and the Services Viewpoint (SvcV). The Technical Standards View was evolved and expanded into the Standards Viewpoint (StdV). The Data and Information Viewpoint (DIV) was created from specific products from the version 1.5 SV and OV. In addition, two new viewpoints, the Project Viewpoint (PV) and the Capability Viewpoint (CV) were added. Just as DoDAF version 1.5 had products supporting views, DoDAF 2.0 has a series of specified models supporting each viewpoint. In addition, DoDAF version 2.0 emphasizes that it is "fit for purpose", meaning that the creation of any particular models or viewpoints is optional depending on the needs of a particular project, and nothing is mandated. Instead, the data is considered the essential element of architecture development [50]. A discussion of supporting models relevant to this thesis is provided in Appendix A.

As was discussed in 1.3.1, it has been suggested that DoDAF can be used in conjunction with the acquisition process. JCIDS mandates the use of certain architecture products at each Milestone to help define the to-be architecture that is expected to be the outcome of the acquisition process. [31] However, there is no mandate for any architecture products to be produced during CBA or pre-Milestone A. Other entities

have suggested that a clear characterization of the architecture could be helpful to conducting CBA. The Air Force Early Systems Engineering Guide recommends that the OV-1, OV-2, OV-3, OV-4, and OV-5 be developed during CBA for the purpose of ensuring that potential system concepts will integrate properly into the SoS [167].

1.3.3 Executable Architecting

The use of executable architecting as a key component of an early phase acquisition process can help to alleviate some of the challenges associated with architecture evaluation. According to DoDAF, an executable architecture is defined as the use of dynamical simulation software to evaluate architecture models [45]. An executable architecture takes advantage of the architecture products and the large amount of information contained in those products to automatically or semi-automatically generate inputs to a dynamic modeling and simulation software [69]. This means that alternate architectures represented by variations in products can be automatically ingested into a simulation environment and information about their performance can be gained. This is particularly desirable since the creation of the products is mandatory, and the products break apart the architecture into perspectives of interest. Thus, variations can be focused within specific products to help isolate the effects of alternatives and reduce the number of cases or combinations that must be evaluated. While this idea is very attractive in theory, there is no existing standard for executable architecting. Some custom software packages exist that often leverage UML as a standard language for representing architecture products and as a standard language for model ingest[99]. A notional figure depicting the idea of an executable architecture is shown in Figure 3 below.

Several advantages of this type of approach make it appropriate for use in an early-phase SoS engineering process, including a decrease in time and effort required to collect needed data, and the ability to actually vary the architecture through the

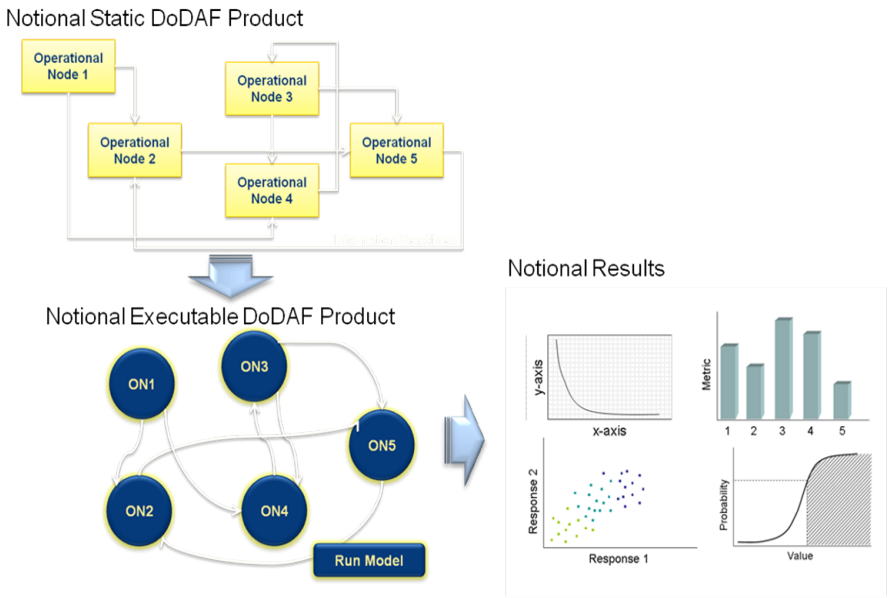


Figure 3: Notional Executable Architecture

products and rapidly analyze and quantify the impacts of those changes. The impact of systems can be assessed within the context of the greater SoS, allowing mission level impacts to be better understood. Furthermore, because the model or simulation is generated from the architecture products themselves, it is required that the architecture products are always kept up to date with the latest changes and revisions, and the architecture itself can be checked for consistency. This type of architecture evaluation can support cost-benefit analysis and help support quantitative acquisition decision-making [69]. However, it is also observed that creating an executable architecting environment is not a trivial task. Choosing modeling and simulation tools that are able to work together and that have the flexibility required to vary all kinds of architecture-level variables (including operational changes, information types and flow paths, system variables, and system interactions) can be very challenging.

From the description above, it is clear that in order to create an executable architecture environment, the following conditions must be met:

1. Architecture products must be standardized, such that anyone attempting to

use the environment will create each product in the same way, and include the information needed to create the desired model(s).

2. Architecture products must be computer-readable. This means that not only must the products be standardized, but the interface to the environment must be able to interpret the graphical, textual, and tabular products.

3. Architecture products must be consistent. Products that share information shown from different perspectives must be reconcilable, meaning that they could be transformed between each other using a set of rules. For example, the OV-2 (showing operational nodes and needlines) should be consistent with the SV-1 (showing the systems resident at those operational nodes and the needlines between them) and the SV-2 (showing how communication across each needline is accomplished). If a needline or operational node exists in the SV-1 or 2 that is non-existent in the OV-2 (or vice versa), there is an inconsistency in the architectural description.

4. The information contained in the architecture products must include the information required by the models, and this information must be presented in a standard way and stored in a standard location.

5. If modeling and simulation tools are to be linked within the executable environment, these tools must be chosen such that they are compatible, or programming must be done that makes the tools compatible.

6. The user of the executable environment must not be ignorant to either the scenario or the modeling tools. Informed decisions regarding which models to use to obtain the information needed for the problem at hand will need to be made. The user must be able to choose the correct subset of models to support decision making for the specific problem.

Although DoDAF 2.0 moved towards a data-centric and computer readable model with the implementation of the DM2, DoDAF is not necessarily designed for executable architecting. In fact, Garcia observed in 2010 that "DoDAF does not lend itself to the M&S [Modeling and Simulation] field, even though M&S would be an invaluable tool to evaluate DoDAF specifications to verify that requirements and objectives are met"[72]. Pawloski et. al. [144] state that the static products of DoDAF, "while capturing enormous amounts of information about the Operational Architecture (OA) and System Architecture (SA), fail to provide a good vehicle for conducting detailed dynamic "behavioral" analysis of how the systems are supposed to interact with each other." DoDAF products do not mandate the collection of much of the quantitative information needed for modeling and simulation, although the architecture products contain the information for the backbone of model creation. However, in order to use DoDAF for executable architectures, metadata must be added to existing DoDAF models to gather the required quantitative information for modeling and simulation.

Mittal [124] provides the following list of DoDAF shortcomings for executable architecting:

1. Although there is mention of "executable architectures" in DoDAF, there is no methodology recommended by DoDAF that would facilitate the development of executable DoDAF models.
2. It has completely overlooked the model-driven development approach. Consequently, there is no formal M&S theory that DoDAF mandates.
3. DoDAF fails to address performance issues at the OV level.
4. DoDAF fails to include measures of effectiveness (MoEs) that can be evaluated at the OV stage. If any performance measures are considered at all, they are at the SV level. System parameters and

performance is at a totally different resolution than MoEs.

5. There is no mechanism to perform verification and validation (V&V) at the OV stage.
6. It fails to address M&S as a potent evaluation and acquisition tool.

The Unified Profile for DoDAF and MODAF (UPDM) has made an attempt to standardize DoDAF products. UPDM was created to "significantly enhance the quality, productivity, and effectiveness associated with enterprise and system of systems architecture modeling, promote architecture model reuse and maintainability, improve tool interoperability and communications between stakeholders, and reduce training impacts due to different tool implementations and semantics" [136]. The UPDM is focused largely on standardizing the representation of the architecture, relying heavily on visual modeling languages such as the Unified Modeling Language (UML) and the Systems Modeling Language (SysML), and has been very successful in this goal. UPDM attempts to leverage existing standards as much as possible, and therefore shares many of the shortcomings of DoDAF and SysML for executable architecting, particularly a lack of executable semantics. While some attempts have been made to make UPDM-compliant architecture representations executable [126], these have been problem specific and have not been implemented in any generalized fashion and have required customized analysis to be built into Architecture-based software packages that support UPDM. Alghamdi et al. have noted that UPDM has not been widely adopted and is not fully mature [8], and thus may not be fully ready to incorporate executable architecting.

Despite the shortcomings of DoDAF for executable architecting, customized modifications and additions have been made by researchers to allow it to be used as a basis for executable architecting with some success. These modifications are discussed more in 3.5.4.

1.4 System of Systems Engineering

1.4.1 Systems Engineering Challenges due to System of Systems Focus

Although SE is a relatively well defined field with several accepted handbooks and standards, such as IEEE 16326 [92] and the INCOSE SE Handbook [89], the DoD continues to face challenges in the execution of SE. The National Defense Industrial Association (NDIA) Systems Engineering (SE) Division Task Group identified the top 5 SE issues in the DoD. The SE Division was formed to "address technical and management issues related to DoD acquisition reform and to promote an integrated and balanced approach to weapon system design" [130]. In 2006, the top five issues identified by the committee included [131]:

- Key systems engineering practices known to be effective are not consistently applied across all phases of the program life cycle,
- Insufficient systems engineering is applied early in the program life cycle, compromising the foundation for initial requirements and architecture development.
- Requirements are not always well-managed, including the effective translation from capabilities statements into executable requirements to achieve successful acquisition programs.
- The quantity and quality of systems engineering expertise is insufficient to meet the demands of the government and the defense industry.
- Collaborative environments, including SE tools, are inadequate to effectively execute SE at the joint capability, system of systems (SoS), and system levels.

These issues represent a general deficiency within the DoD in the early phases of SE and requirements identification. The committee's note of a general inadequacy to effectively perform SE at the SoS level underscores the need for a standardized

process that goes beyond the guidance provided by JCIDS and DoDAF to increase the reliability of SoSE efforts within the DoD.

While many argue that the distinction between a system and an SoS (and therefore SE and SoSE) is simply a matter of perspective and scope (i.e. to an engine designer, the engine would be the system and the aircraft would be the SoS, but to an aircraft designer, the aircraft would be the system and the National Airspace System (NAS) would be the SoS), this work differentiates an SoS from a system by several key features, that are particularly relevant for defense SoSE. The first distinction is seen at a managerial and funding level. In a system, there is often a clear and predefined set of stakeholders that are intending to procure and use the product. In an SoS, a broader range of stakeholders are included. These stakeholders include both the stakeholders of the individual systems and the stakeholders for the overall SoS. Additionally, in an SoS, it is rare that a single source of funding will span the entire SoS. In fact, it is much more likely that individual systems will be funded by a diverse group of organizations, and the goals of the funding sources for individual elements may not align with the goals of the SoS stakeholders. This presents significant challenges in the SoSE process, because of potentially conflicting and ever-changing requirements [57].

This is closely related to the second distinction between a system and an SoS. In a system, the piece parts (subsystems) of the whole are designed to fulfill a specific role in the system, with the intention that these subsystems will be designed specifically to help meet the goals of the overall system. In an SoS, however, the individual piece parts (systems) are not necessarily designed to specifically contribute the goals of the overall SoS. In fact, these systems often have a unique and independent purpose from the overall SoS. This means that the role they are playing within the SoS is not necessarily the role for which they were designed [57]. Therefore, the SoSE process must account for elements which may not be designed to specifically fill the role in

which they are being used. Furthermore, there is a distinct difference in the way in which boundaries and interfaces are considered between a system and an SoS. In a system, the interfaces between subsystems are an inherent part of the design, and the external boundaries and interfaces only include those of the specific system being considered. In an SoS, there is a significant focus on data, information, and resource flow between systems that were not necessarily designed with the intention of interfacing them, presenting a significant challenge to the SoSE [57].

A third distinction lies in the area of requirements and performance assessment (including testing and evaluation). In a system, requirements and performance goals are typically given with specific performance objectives. Testing and evaluation is focused on verifying that the system meets these specific objectives. The requirements of an SoS focus on meeting capability needs or improving mission performance, which are often enumerated in nebulous statements not associated with specific, measurable objectives. Testing and evaluation becomes very difficult, not only because of nebulous requirements and performance criteria, but also because of the sheer size and number of systems included in the SoS. This challenge is further exacerbated by the various life cycles of the systems in the SoS. Not only are the life cycles different for the systems, the systems are also in different phases of their life-cycles. Some are legacy systems, some are modified or technology-infused systems, and some are altogether new systems [57]. These differences cause a variety of testing and evaluation challenges in SoSE.

1.5 Observations and Primary Research Objective

Several key observations can be made from the discussion in the previous sections. These observations motivate the driving research question that inspires this research, and provide guidance on what beneficial characteristics should be included in an SoS architecting method.

1. The early phases of systems engineering and architecture selection are where the most fundamental (and most difficult and costly to reverse) decisions are made
    (a) JCIDS and DODAF make this more difficult by providing static products [69]
    (b) In the defense industry, more systems engineering is needed early in the program life cycle, to create the foundation for initial requirements and architecture development [131]
    (c) Effectively translating capability needs into executable requirements is important to successful acquisition, and is currently a struggle within the defense industry [131]
2. There is not a standardized method for moving through the JCIDS process
    (a) Moving through the development and acquisition process is not trivial
    (b) JCIDS provides a checklist of documentation and approval phases that support capability based acquisition
    (c) Key decisions must be made in order to support the creation of these documents
3. There is not a standardized method for creating the views that are required to create DoDAF products, or for using the information contained within them to support early-phase acquisition decision making
    (a) DoDAF provides a checklist of views and products that support system development
    (b) Key decisions must be made in order to support the creation of these products, and some of the products are required at Milestones A and B

(c) Creation of DoDAF products, even for the baseline, requires a multi-disciplinary understanding of the problem and the collection of a significant amount of information on organization, operational needs, information requirements, systems, their performance, and their interactions

(d) If the information within the architecture description could be used to not only create the products themselves, but also to understand the problem, develop the requirements and the trade space, and spawn the creation of new alternatives, the time and resource investment required to create the DoDAF products can be reused, and more information may become available earlier in the acquisition process

These observations inspire the primary objective of this research, which is stated below.

Primary Research Objective: *To create a capability-based systems engineering method for the early phases of design and acquisition (including gap analysis and alternative evaluation) which improves agility in defense acquisition by (1) streamlining the development of key elements of JCIDS and DODAF, (2) moving the creation of DODAF products forward in the defense acquisition process, and (3), using DODAF products for more than documentation by integrating them into the problem definition and analysis of alternatives phases.*

Although most of the terms used in this objective have been defined previously, there are two that deserve clarity. Agile is defined by the Webster dictionary to mean "nimble; deft" [107]. In the context of defense acquisition, agility refers to the ability to navigate through the processes and requirements more quickly and with increased confidence. Streamlining is defined by Webster's dictionary to mean "make with a form minimizing air or water resistance" or "purge of unnecessary elements" [107]. It is the second definition that is relevant here. In the context of this research, streamlining means to develop only those elements which are necessary, to develop

them in the simplest and most straightforward possible way, and ensure that they contain all of the information that is needed, and nothing more.

This research proposes a way to meet this objective in the form of the ARCHITECT Method, and shows an example of the application of this process to a proof-of-concept example problem. In order to evaluate the success of this method, several observations are made as to both what is required by the acquisition guidance for CBA as well as critiques of the CBA process that should be addressed in a new methodology. From the information in the previous sections, it is suggested that the following characteristics are desirable in a CBA methodology:

1. The methodology should allow CBAs to be conducted more quickly (less than a year)

2. The methodology should result in a CBA that is transparent

3. The methodology should provide decision makers with an increased number of alternatives across the DOTMLPF spectrum

4. The methodology should allow materiel solutions to be evaluated with respect to multiple missions

5. The methodology should leverage quantitative analyses when possible

6. The methodology should be rigorous and repeatable, but have enough flexibility to apply across a broad spectrum of problems

7. The resulting CBA should include a dynamic environment that allows decision makers to interact with results (similar to an interactive design review)

This research attempts to address some of these concerns, particularly the time required to implement parts of the process, the need to evaluate solutions across capability and mission areas, and the need to use a rigorous, traceable, repeatable method

that utilizes modeling and simulation to better substantiate early-phase acquisition decisions. This research specifically focuses on the CBA conducted during the Pre-Milestone A phase of acquisition, and how the information from the CBA can better support Milestone A decision making. The research does not focus on Post-Milestone A activities. However, the author recognizes that many of the challenges of CBA are rooted in political and social institutions and processes, and cannot be addressed through a methodology. It is not the goal of the research to address every concern or criticism of CBA. The methodology presented here was developed with these concerns in mind, and design of the methodology and the techniques selected to support each step of the methodology are chosen based on their potential to mitigate these concerns. It is the goal of this research to develop a methodology that works within the current guidance and processes and could potentially help to improve CBA results and help those leading CBA studies to mitigate these challenges more easily. This research will attempt to address as many of the concerns as possible at the methodology level, but does not comment on the implementation of the methodology within DoD, recognizing that there are organizational challenges outside of the control of this research that would also have to be addressed.

It is the goal of this research to develop a methodology which is distinct from current CBA practices in several ways. First, the ARCHITECT methodology will attempt to address both the *what* (i.e. what information needs to be developed, this is based off of the existing guidance) and the *how* (i.e. what techniques are available to help obtain the needed information, and which should be used when) for CBA. Furthermore, it utilizes DoDAF more extensively than suggested by current CBA guidance, and uses it for more than just documentation. In fact, the architecture framework is used as a key enabler for exploring and modeling the alternative space. By doing this, the ARCHITECT methodology is able to explore a wider variety of solutions than have been historically considered in CBA, including a stronger focus

on non-materiel solutions. The methodology aims to increase the use of quantitative analyses over traditional early-phase studies, and to increase the rigor with which all steps of the CBA are performed.

CHAPTER II

DEVELOPMENT OF RESEARCH QUESTIONS

The goal of this chapter is to develop specific research questions that, if answered, address the primary research objective stated in the previous chapter. In order to do this, it is first necessary to characterize in general terms the problem that is being addressed, and then decompose this problem into relevant elements that can be studied. Each of these elements is then mapped to related fields to identify key areas that can be researched to inspire research questions and guide the search for answers. Although the research objective presented in the previous chapter pertains specifically to defense acquisition in an SoS context, there is a broader and more fundamental theme underlying the research to answer this objective, which is, broadly stated, acquisitions in an SoS context. Thus, despite the limited availability of academic literature on defense acquisition practices, other fields have studied aspects of this broader problem, and this work can be leveraged and applied to guide the development of defense acquisition methodologies.

Considering first the general topic of acquisition, there are four primary elements to the acquisition problem. First, there are the decision-makers and SMEs as individuals. These individuals hold unique perspectives and experiences that can bring both useful insight and cognitive biases to the acquisition decision-making process. Within cognitive psychology, the cognitive strengths and biases brought to bear by an individual performing strategic decision-making has been studied in depth. There are several works of particular noteworthiness in relation to acquisition decision-making. Schwenk and others have explored the biases of decision-makers in acquisition, and the impact of these biases on decision-making outcomes in [60, 155, 154]. Busenitz et al.

have studied the strategic decision-making processes of managers and entrepreneurs, and the level to which each is subjected to biases in [27]. These references provide an overview of the research done in these areas. Hammond et al. provide an overview of hidden decision-making traps in [80]. This research can be leveraged in the defense acquisition methodology development to reduce cognitive biases and leverage the positive benefits of using the past experience and knowledge of decision-makers and SMEs where appropriate, and is discussed further in this chapter.

Second, these individuals then must interact in a group to make decisions, which brings the element of group dynamics into play. Group dynamics on the whole are studied in the field of organizational psychology. However, given that group decision-making is such a prevalent occurrence in engineering, business, and government, the field of decision theory and has emerged. Within decision theory, multi-agent decision-making explores the challenges of having multiple players with differing goals and perspectives trying to reach a consensus. Other branches of decision theory (such as multi-attribute or multi-criteria decision-making) study how to make decisions in the presence of multiple and conflicting criteria, which is an important factor in the multi-agent decision-making as well. Arrow is perhaps the most well known scholar in this field, and his work studying the biases imposed by different ways of combining decision-maker preferences led to the formulation of Arrow's impossibility theorem, which is further discussed in [11], and which has been the basis for much work in the area of group decision-making. Black has proposed a general scientific reasoning theory for group decision-making across a wide range of fields, including politics, government, and acquisition decision-making. This is outlined in [18]. This research area can be leveraged to determine how to best support compromise and the reaching of consensus within defense acquisition decision-making. As group decision-making will not be a focus of this work, these ideas will not be discussed in detail here. Rather, the inclusion of group decision-making is left as future work.

Third, decision makers and analysts go through some process to identify a problem, identify potential solutions, compare these solutions and make a decision. This process element has been studied within the field of management science, and in particular with respect to corporate acquisitions. This research has examined what process elements should be included, the order in which they should be executed, and the characteristics of performance for the elements that lead to a higher rate of success in a corporate acquisition environment. Jemison and Sitkin have emphasized the importance of process in acquisition decision-making, and discussed a generalized process perspective in [97]. Singh and Montgomery [157] have reviewed the corporate acquisition process and how the process and the resulting decisions impact economic performance of the corporations. Ullman [164] has been a thought leader in the area of developing sound principles for decision support systems to support the execution of these processes. While there is not a one-to-one parallel between corporate acquisition and defense acquisition, the underlying principles of process and management are similar, and this research can be leveraged to help determine the process element of the defense acquisition methodology. This is discussed further in this and the following chapter.

Finally, the decision makers depend on information to support the execution of the decision-making process. As the decisions can only be as good as the information used to make them, obtaining useful, relevant, and sufficient information is critical to decision-making success. Engineering as a whole is concerned with analysis to gain information and insight about systems and solutions. Engineers have extensively researched the creation of models and the testing of existing systems to gather information needed to make decisions about both new and existing systems. Systems engineers in particular have focused on the full life cycle of a project, starting from the earliest phases of requirements definition, to improve the quality of information available to make decisions throughout the system life cycle. In recent years, the field

of SoSE has developed to extend these practices beyond a single system to the SoS as a whole. Modeling techniques have been developed by many engineering disciplines to model and address a wide range of potential types of systems and processes. As it is necessary to pull from a broad range of engineering fields, there is no one reference which will cover the basics of all the necessary topics. Thus, to address the information element, many fields of engineering can be explored to identify potential techniques for supporting the gathering of information in the context of defense acquisition decision-making. These are discussed in detail in Chapter 3.

With these elements of acquisition in mind, there are additional challenges brought to bear when performing acquisition decision-making in an SoS context. The SoS problem is characterized by a large combinatorial space combining ways and means into useful capabilities. As was discussed in the previous chapter, this results in additional complexities and challenges when analyzing an SoS or a system's integration into an SoS as compared to a single system isolation. Two of these challenges are particularly pertinent in acquisition. First, the analysis is complicated by the large number of alternatives, the complex interactions between elements, and the need to account for process and human factors in addition to physics when modeling the SoS. Second, the acquisition decision-making environment is characterized by a larger number of stakeholders, where often, many of the stakeholders have individual interest in an element of group of elements rather than the SoS as a whole. These challenges correspond to the information element of the acquisition decision-making problem, and the group dynamics element of the acquisition decision-making problem, respectively.

Figure 4 summarizes the four identified elements of acquisition decision-making and the academic fields identified as being most relevant. Two of these elements, group dynamics and data and information gathering have been identified as having unique challenges when performing acquisition decision-making in an SoS context. For the purpose of this thesis, the overall process and process flow of the methodology

will be developed by leveraging research from the defense acquisition community (where available) and from the management science community, particularly in the area of corporate acquisitions. Cognitive psychology research will be leveraged to help identify potential biases, and tailor the methodology to mitigate these biases wherever possible. These two elements will be discussed in this chapter. A strong focus will be placed on how to gather the data and information to support acquisition-decision making in an SoS context, and literature from a broad variety of engineering and science disciplines will be leveraged to address data and information collection. The supporting literature search for this element will be presented in the following chapter. Finally, while the methodology will recommend that decision theory be applied when using the methodology, the challenges posed by group dynamics in decision-making will not be explicitly addressed by this thesis. It is the intent of the author that the methodology developed here will provide the decision makers with a process which reduces individual biases and helps to gather the large amounts of information needed for SoS acquisition decision-making. These individuals would then be able to come together as a group with a common information set, and in this setting multi-agent decision-theory principles could be applied to reach a final consensus. The unique challenges of group decision-making in this context will be left as a topic of future work.

This perspective on acquisition decision-making is similar to the perspective on decision-making in general presented by Bell and Raiffa in [16]. They have broken decision-making in general into three areas: the logical, the methodological, and the psychological. These correspond to the information, the decision-maker as and individual, and the process in the breakdown presented here. However, they deal primarily with decisions made by a single individual, and thus their framework does not include the impact of the group on decision-making. It is worth noting that the fields listed here are not a complete listing of all seemingly relevant fields, nor all

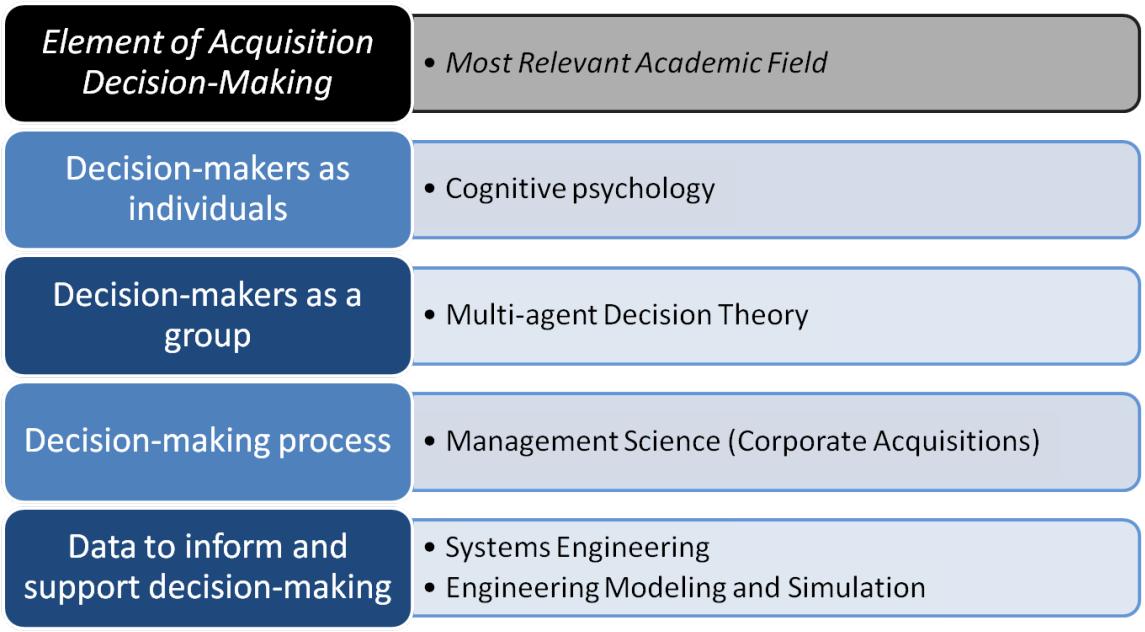


Figure 4: Summary of Acquisition Elements and Related Fields

of the fields examined for potential inclusion in this research, but rather those that the author feels are most applicable to the problem at hand. For example, the field of operations research may seem at first glance to be very relevant. However, the level of detail typically examined in operations research and the information typically used in operations research studies exceeds that which is expected to be available in early-phase defense acquisition decision making. That is not to say that some aspects of operations research cannot be leveraged for development of this methodology. In fact, some of the modeling and simulation techniques that will be discussed in the following chapters are commonly used in operations research studies. However, at the most fundamental level, operations research solves a slightly different class of problems at a different level of detail than that being considered here, and thus is not used as one of the guiding fields.

2.1 Perspectives from Corporate Acquisitions

A rich body of research exists on strategic acquisition decision making processes for corporate acquisitions. Many strategic decision making processes have been proposed.

Schwenk [155] has performed a survey of existing models and developed a derived model that highlights the key steps common across all processes. These include goal formulation, problem identification, alternatives generation, and evaluation and selection, and implementation [155]. It can be noted that these steps are similar to the steps laid out in the CBA process in [101]. Further exploration reveals that there are many similarities between corporate acquisition and defense acquisition, including:

- Acquisition is usually performed by large organizations with diverse stakeholders and many "missions" or lines of business [97]
- Organizational complexity is a characteristic of the acquisition process. [155, 97, 27]
- Decisions are typically made in face of ambiguity and uncertainty. [155, 139, 27]
- In particular, there is a large amount of environmental uncertainty that cannot be minimized by organizational action. [97]
- In general, strategic decision-making involves several key activities, including goal formulation, problem identification, alternatives generation, and evaluation and selection. These are the same activities as take place in defense acquisition. [155]

For corporate acquisitions, there are several perspectives on acquisitions. Much of the historical research on acquisition uses a rational choice perspective. This perspective assumes that the acquisition decision maker is a rational decision maker. This perspective examines the 'strategic fit' and the 'organizational fit' as elements in the acquisition decision process. The strategic fit refers to the degree to which two businesses that are to be merged complement each other, and the degree to which the candidate for the merger contributes to both the financial and non-financial goals of the parent company [97]. This can be thought of as the potential value of the

acquisition. The organizational fit refers to the match between the administrative and cultural practices of two organizations, as well as the match between personnel and day-to-day operations [97]. This can be seen as the ease of integration of the acquisition.

However, it also been recognized that many acquisitions do not yield the expected returns, and it has thus been suggested that the rational choice perspective is an incomplete view of acquisition [98, 113, 97]. In response to this, Jemison and Sitkin [97] developed the process perspective on acquisition to supplement the rational choice perspective. The process perspective includes consideration of the acquisition process itself as an influence on the success of the acquisition. Jemison and Sitkin [97] note that, "Acquisitions are strategic, complex, occur sporadically (for most firms), and affect varied stakeholder groups and multiple actors whose involvement is temporally and functionally divided. These factors, in combination, result in an acquisitions process that is both discontinuous and fractionated." Several experts in the field suggest that improving the acquisition process itself can improve the success rate for corporate acquisitions, particularly in cases where collaborative operational interaction is required between multiple firms [97, 139, 83] It can be noted that the same characteristics used to describe corporate acquisitions can be said to apply to the landscape of defense acquisition as well (as noted above), and it can therefore be hypothesized that similar struggles will occur in defense acquisition. If this is true, it then follows that improving the acquisition process would improve the success of defense acquisitions as well as corporate acquisitions.

Jemison and Sitkin suggest four main ways in which process may impede acquisition success: activity segmentation, escalating momentum, expectational ambiguity, and management system misapplication. Activity segmentation refers to the subdivision of roles and responsibilities as a result of the complexity of analyses, the large number of tasks, and the various roles of specialists. Jemison and Sitkin argue that

this segmentation results in isolated analyses that use inconsistent perspectives and easily quantified analyses, which are then poorly integrated and lead to a disproportionate emphasis on the strategic fit over the organizational fit [97]. The impact of activity segmentation on acquisition is shown in Figure 5.

Escalating momentum refers to the waxing and waning pace of acquisition between periods of waiting and activity, with the activity periods being characterized as frenzied. In general however, it has been observed that there is an overall increasing pace and desire to complete the process more quickly, leading to potentially premature solutions and less consideration of integration. Several characteristics have been identified as contributing to the escalating momentum, including participant commitment, secrecy, decision-maker isolation, over-confidence, decision-making under conditions of ambiguity, self-interest of the participants, and resistance of the target firm to the acquisition attempt [97]. The impact of escalating momentum on acquisition is shown in Figure 6.

Expectational ambiguity refers to the ambiguity experienced during the negotiation phase of an acquisition that is then carried into the integration phase. When ambiguity carries from the negotiation phase into the integration phase, disagreements between stakeholders on the key points of an agreement may become a problem. When ambiguity in the requirements or the integration plan result in conflict between stakeholders, polarization may occur. This can lead to hostile feelings and actions on the part of both organizations, which can undermine acquisition success [97]. The impact of expectational ambiguity on acquisition is shown in Figure 7.

Finally, management systems misapplication refers to the often heavy-handed application of the parent company's strengths to the subsidiary without utilizing the original plan to merge the strengths of both companies. This is attributed to a combination of defensiveness and arrogance on the part of both companies. The result of this is that the parent company imposes its own practices uniformly on

the subsidiary without regard for the potentially more appropriate practices of the subsidiary, and the employees of the subsidiary may respond with resistance to the new policies [97]. The impact of management systems misapplication on acquisition is shown in Figure 8.

While not all aspects of these four process pitfalls are relevant to defense acquisition, there are several relevant lessons about what can improve the probability of success for a acquisition program. There is an analogy between the strategic fit of a company and the expected performance/value of a DOTMLPF alternative. The organizational fit can be seen as analogous to the implementation or integration difficulty associated with a particular DOTMLPF alternative. Using this analogy, it can be observed that integration of relevant information and a more balanced focus on both performance of a potential acquisition alternative as well as its ability to integrate with the existing framework are both important elements to successful acquisition. Additionally, it is important that analysis be done with a similar scope and perspective, and a consistent set of assumptions between analysts. Furthermore, it can be expected that escalating momentum occurs as organizations are under pressure to meet acquisition milestones in a timely manner and to secure needed funding as quickly as possible. Thus, it is to be expected that premature solutions and an underemphasis on integration of new solutions occur. Although there is not a negotiation phase similar to the negotiation experienced in corporate acquisitions, it has been noted that there is often requirements ambiguity in the early phases of defense acquisitions [131]. It should be expected that this ambiguity will lead to sub-optimal solutions and disagreements between stakeholders as to which solutions should be pursued. Finally, while defense acquisition does not typically result in a change of management, it is important to integrate the new solution into the existing framework in such a way that the expected benefits of the solution can be realized. From these observations, several requirements for a successful defense acquisition process

are recognized:

- Integration of information and data relevant to decision makers is important, and should not be done in an ad-hoc fashion
- Scenarios, assumptions, and baseline information should be consistent among all analyses
- The process must include sufficient analysis and full consideration of the alternative space and be completed in a short time frame
- Clarifying requirements and reducing ambiguity is important to successful acquisition
- Emphasis should be placed on ease of integration of solutions as well as performance and value
- It should be verified prior to choosing that solution that a new solution can be integrated in such a way that expected benefits are realized and integration challenges will not hinder success

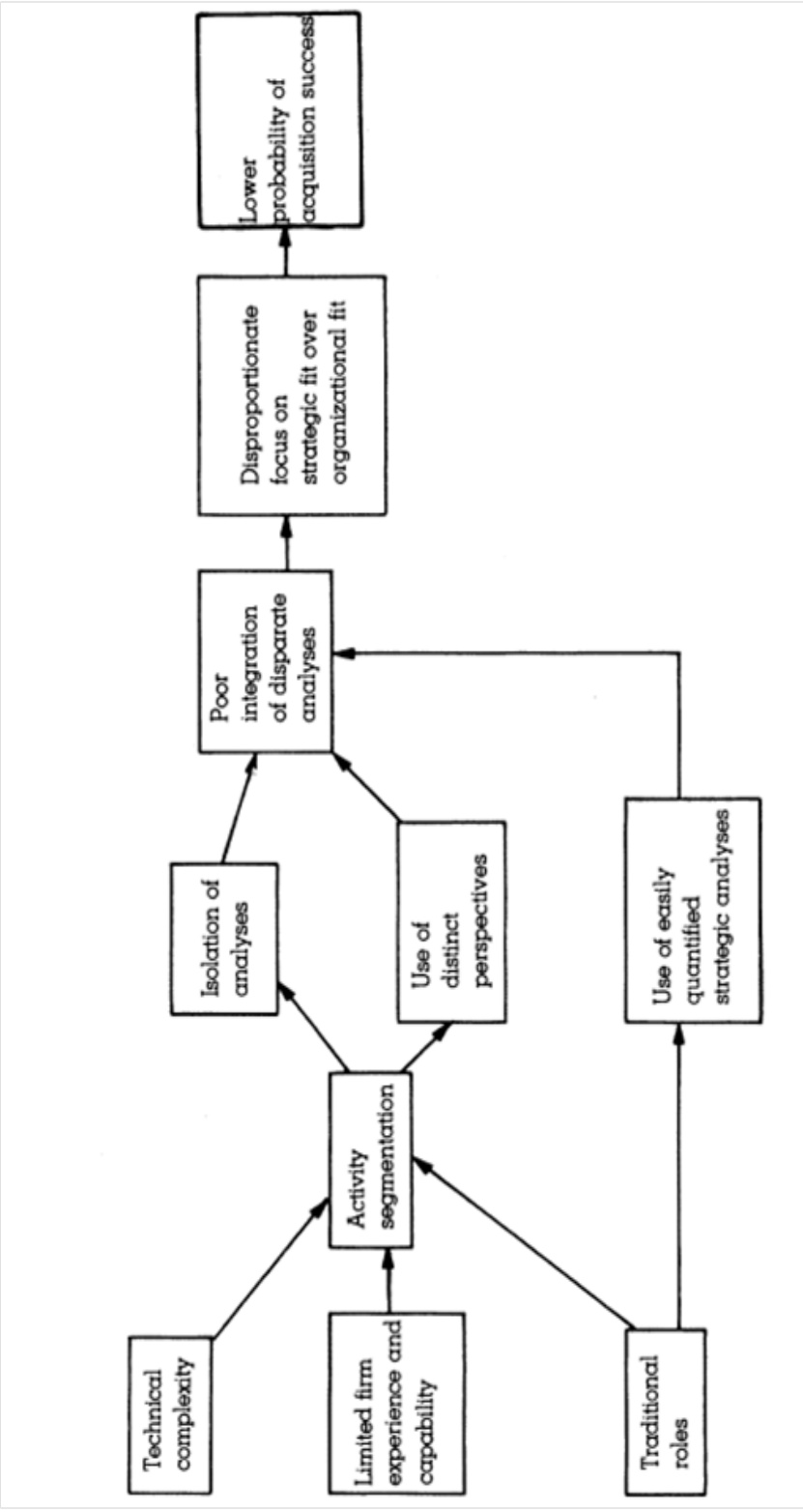


Figure 5: The Influence of Activity Segmentation on Acquisition, according to Jemison and Sitkin. Reproduced from [97]

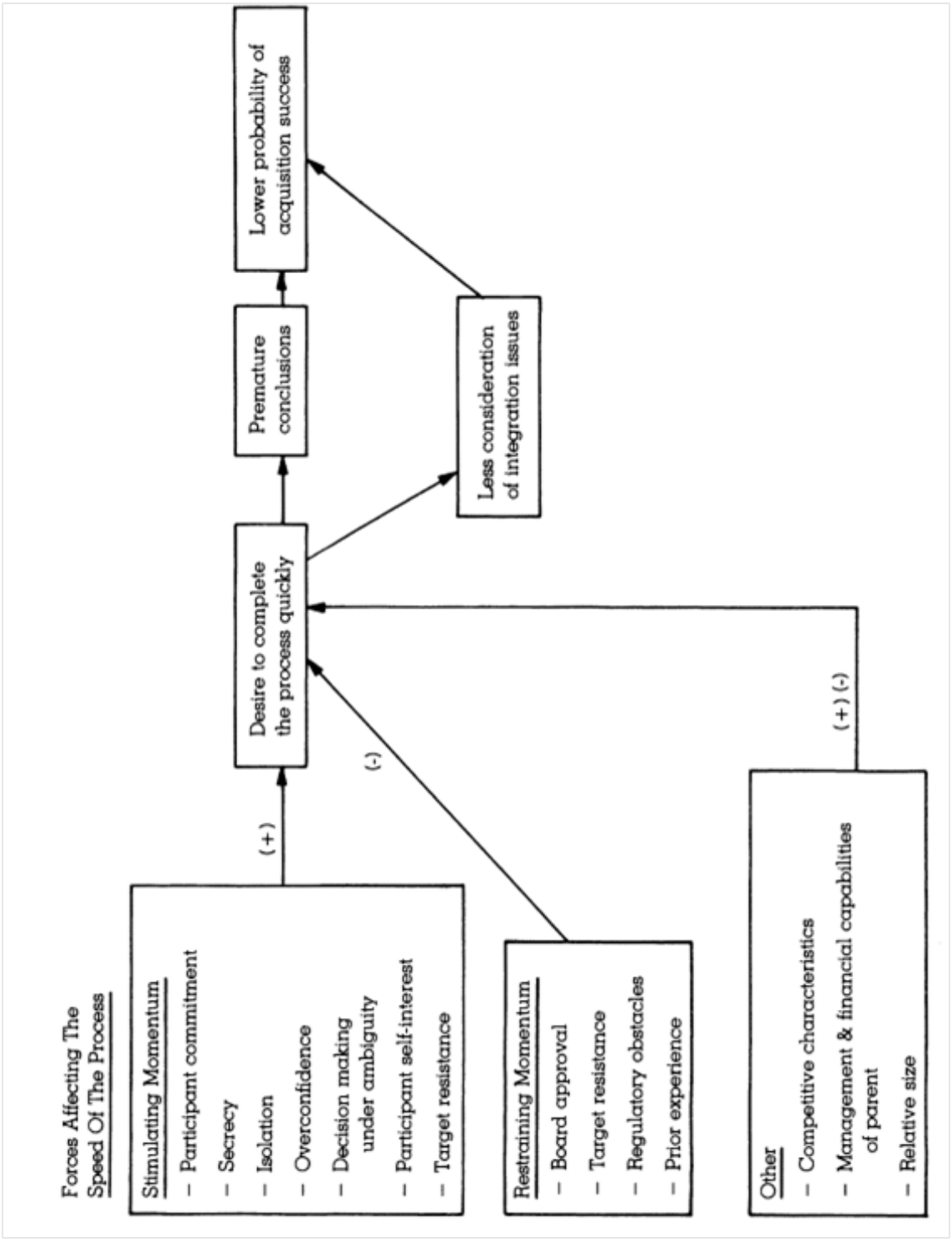


Figure 6: The Influence of Escalating Momentum on Acquisition, according to Jemison and Sitkin. Reproduced from [97]

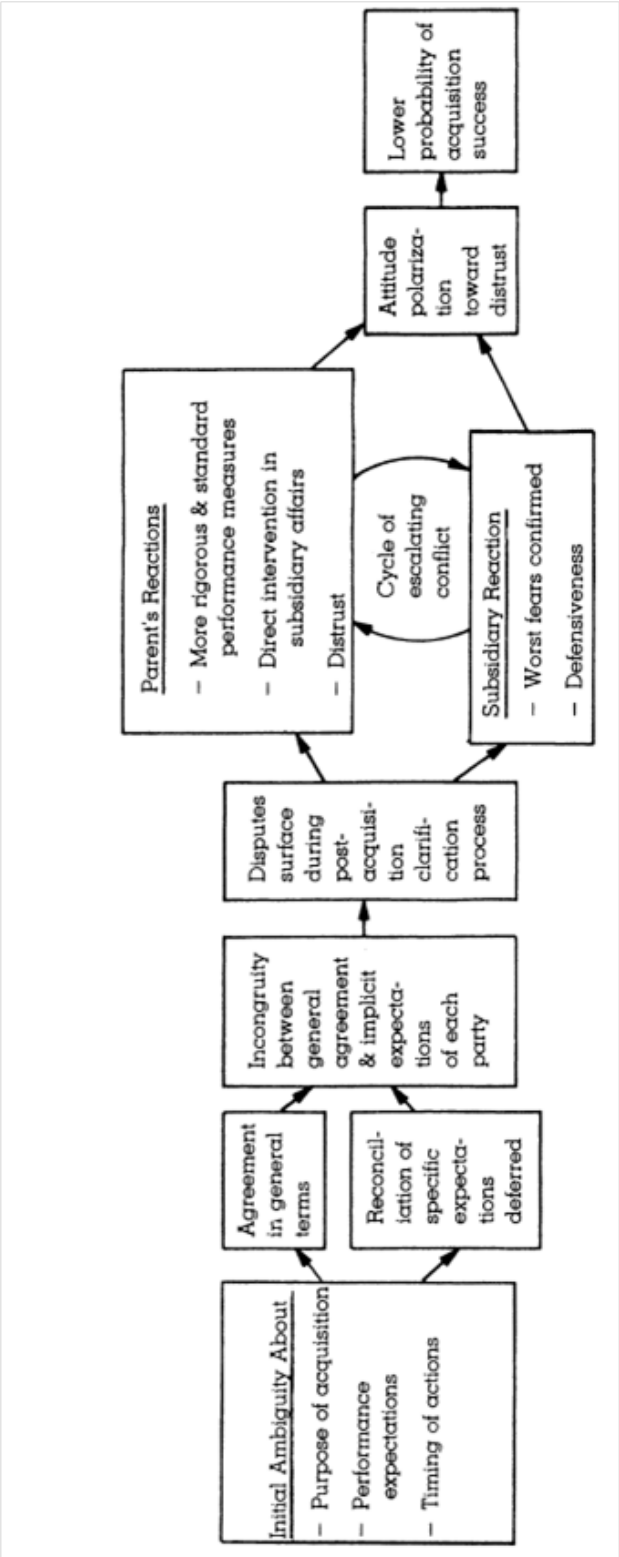


Figure 7: The Influence of Expectational Ambiguity on Acquisition, according to Jemison and Sitkin. Reproduced from [97]

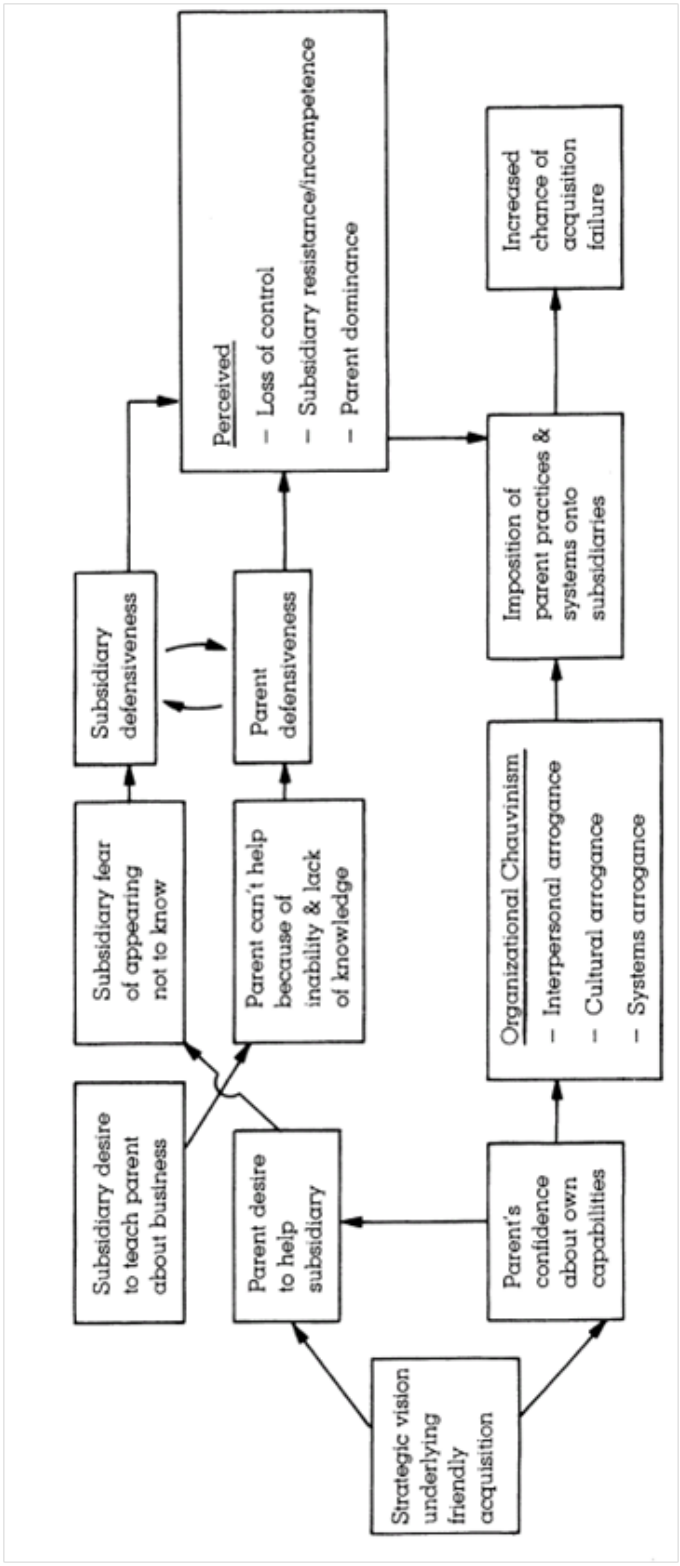


Figure 8: The Influence of Management Systems Misapplication on Acquisition, according to Jemison and Sitkin. Reproduced from [97]

2.1.1 Cognitive Simplification Processes in Strategic Decision-Making

Acquisitions in general have been conceptualized as simply being decision-making processes [139]. However, as stated previously, this decision-making is often done with little structure in the context of extreme complexity ambiguity, with a high level of uncertainty. Since human processing capacity is not infinite, researchers in cognitive psychology and behavior theory have identified a series of cognitive simplification processes that decision makers use during the acquisition cycle and which steps of the process these affect. Schwenk has performed a survey of these simplification processes. In addition, he has presented a series of laboratory experiments and case studies demonstrating the potential negative impact of these cognitive simplifications [155]. Understanding these potential pitfalls can help develop an acquisition decision making process that aids users in avoiding these pitfalls and in managing the cognitive complexity of the decision environment. Schwenk's findings will be summarized here, followed by a discussion of the potential impact of these findings on the development of acquisition decision-making processes. A series of tables summarizing the cognitive biases presented by Schwenk are shown in Tables 1 through 3.

In the area of goal formulation and problem identification, Schwenk identified four potential simplification processes. Prior hypothesis bias refers to the tendency of individuals to form erroneous or poorly supported beliefs about the relationships

Table 1: Summary of Cognitive Simplification Processes and their Effect on Goal Formulation and Problem Identification. Recreated from [155]

Stage I Goal Formulation/Problem Identification	
Process	Effect
(1) Prior Hypothesis Bias	(1) Evidence ignored, gaps not perceived
(2) Adjustment and Anchoring	(2) Evidence under-used, gaps not perceived
(3) Escalating Commitment	(3) Significance of gap minimized, strategy not revised
(4) Reasoning by Analogy	(4) Problem mis-defined (oversimplified), inappropriate strategy revision

Table 2: Summary of Cognitive Simplification Processes and Their Effect on Strategic Alternatives Generation. Recreated from [155]

Stage II Strategic Alternatives Generation	
Process	Effect
(1) Single Outcome Calculation	(1) Restricts alternative to a single one
(2) Inferences of Impossibility	(2) Premature rejection of alternatives
(3) Denying Value Trade-offs	(3) Biased use of evaluation criteria
(4) Problem Sets	(4) Alternatives restricted

Table 3: Summary of Cognitive Simplification Processes and Their Effect on Evaluation and Selection. Recreated from [155]

Stage III Evaluation and Selection	
Process	Effect
(1) Representativeness: (a) Insensitivity to predictability, (b) Insensitivity to sample size, (c) Illusion of validity	(1) Inaccurate prediction of consequences of alternatives
(2) Illusion of control	(2) Inaccurate assessment of risks of alternatives
(3) Devaluation of partially described alternatives	(3) Rejection of strong but poorly presented alternatives

between variables, and then make decisions based on these beliefs rather than abundant evidence to the contrary. Adjustment and anchoring comes about a result of the process of making initial judgments and revising these judgments as more information becomes available. It has been noted that in these situations, the adjustments in judgment are often less than the evidence would suggest, resulting in final estimations that are biased toward the original, less-informed estimations. Escalating commitment refers to the tendency of individuals to be less likely to recognize gaps once significant resources have been invested in a project. This results in decision-makers remaining with a failing project in spite of evidence that it is not productive. In fact, Schwenk states that "Researchers have found that once an individual commits significant resources to an investment project, he will tend to allocate more to the project if he receives feedback indicating that the project is failing than if he receives feedback indicating that it is succeeding" [155]. It was found that escalating commitment was more likely in cases where decision-makers had a risk of losing their jobs and in cases where there was a strong organizational resistance to the chosen course of action. Reasoning by analogy occurs when decision-makers apply simple analogies or analogies from less complex situations to the more complex strategic decision space. While this helps to reduce the perceived level of uncertainty and has been shown to help generate more creative solutions, this perceived reduction in uncertainty can be misleading and result in an oversimplified view that is not representative of the real problem. The failure to recognize the extent to which the analogy actually represents the real problem can lead to an incorrect or oversimplified view of the problems and gaps [155]. A more detailed discussion of this group of biases is provided by Hammond in [80].

In the area of alternative generation, several cognitive simplifications have been

identified that result in a prematurely reduced alternative set and biased use of evaluation criteria. Single outcome calculation is the tendency of individuals or organizations to focus on a single objective and a single alternative rather than exploring all relevant objectives and a broad range of alternatives. In fact, it has been noted that probabilistic models of outcomes are often replaced with assumptions about which alternatives would have favorable and unfavorable outcomes based on the decision-makers pre-defined preferences, resulting in a single-valued problem with a single alternative. This simplification process is more likely to occur when decision environments have a high degree of uncertainty and complexity. Next, inferences of impossibility suggests that decision-makers may rule out non-preferred alternatives by focusing strongly on the negative qualities of an alternative and convincing themselves that this alternative would be impossible to implement. This results in a biased elimination of potentially good alternatives. Denying value tradeoffs, like the previous two, occurs when a decision-maker has a preferred alternative in mind. In this simplification process, the decision-maker interprets an alternative to have high values in many criteria with little cost, ignoring any tradeoffs with this alternative. Last, problem set refers to a decision-making process in which a single problem solving strategy is used, making it harder and less likely for other problem solving methods to be implemented. This can include using a single set of assumptions repeatedly and not revisiting the assumptions in a new environment or scenario [155].

In the area of evaluation and selection of alternatives, three cognitive simplification processes are identified. First, representativeness, which is a bias that stems from a decision-makers tendency to over-estimate the degree to which a case study, model, or data set is representative of the scenario to which he is generalizing it. This is often seen when decision-makers use simple analogies to justify strategic decisions. There are three elements that contribute to this bias. An insensitivity to predictability means that decision-makers to not account for the degree to which the evidence is

relevant or reliable. An insensitivity to sample size can occur when decision-makers do not have a sufficiently-sized data set with which to generalize, but generalize anyway, with the belief that these generalizations are still accurate. This is sometimes referred to as 'the law of small numbers'. This is particularly evident when there are one or a handful of particularly clearly described cases. The third cause of representativeness is the illusion of validity, which occurs occurs when decision-makers are unaware of the uncertainty and inability to forecast in complex decision environments. This causes decision-makers to be overly confident in predictions and decreases emphasis on formulating contingency plans. The second simplification process in the area of evaluation and selection is the illusion of control. This occurs when decision-makers overestimate their personal influence on acquisition outcomes and therefore overestimate the degree to which risks can be mitigated by personal effort. This leads to an underestimation of potential risks to a particular alternative. Finally, the cognitive simplification process of devaluation of partially described alternatives refers to individuals' preference to better described alternatives, because there is a perception of less uncertainty. Thus, poorly described alternatives tend to be discarded more easily despite potentially good performance [155].

While these simplification processes have been shown to be potential pitfalls for decision makers, it is also necessary for the decision makers to simplify information in order to make sense of complex and uncertain acquisition environments [155]. Therefore, it necessary to explore ways to simplify the presentation of information to decision makers while helping to avoid these pitfalls. An acquisition methodology therefore should consider the potential cognitive biases and attempt to help decision-makers reduce the effect of these biases and improve decision-making results.

2.2 Existing Methods for Defense Acquisition

There is a conspicuous void in the open literature in the area of methodologies for conducting the analyses required by defense acquisition, and in particular for conducting CBA. Although CBAs are regularly conducted, they are conducted by a diverse range of organizations, and the methodologies are not formalized or documented. Furthermore, as the CBAs themselves often contain sensitive information, the CBA reports typically have restricted access. Thus, most of the information regarding the successes and detriments of previous CBAs comes from the CBA Handbook and from General Accounting Office (GAO) reports. The only two published methodologies that claim to be a methodology to support acquisition are the Defense Acquisition Program Support (DAPS) Methodology and a method published by the United States Air Force in the Early Systems Engineering Guidebook, which will be discussed briefly below.

2.2.1 Defense Acquisition Program Support (DAPS) Methodology

The Defense Acquisition Program Support (DAPS) Methodology was designed to support the Program Support Review (PSR) Process begun in 1994 by providing "the tailorable framework for conducting PSRs to assist program managers and DoD decision makers in preparation for milestone decision reviews" [133]. An overview of the DAPS Methodology, as presented by [133], is shown in Figure 9. However, like other defense acquisition guidance, the DAPS Methodology describes *what* should be done as program support for each area of the acquisition process, rather than *how* things should be done. In this way, it is a different type of methodology than that presented in thesis, but a short discussion of the methodology will be provided in order to highlight several key points.

DAPS has five major focus areas, including Mission Capabilities, Resources, Management, Technical Process, and Performance. Each of these areas has several sub-areas, and each sub-area has several factors. For each factor, a list of criteria is

provided for that factor, as well as a list of questions to be answered which are referred to as focus areas. While the concept of providing criteria and focus areas is generally desirable, with almost 60 factors and multiple criteria and focus areas under each factor (sometimes as many as 40-50 of each per factor), the amount of information required to comply with DAPS is monumental. Furthermore, it is unclear when certain criteria do and do not apply. For example, for the factor 'Systems and Software Engineering (SSE) Tools', under Pre-Milestone A, one of the listed criteria is "Engineering design is supported by the use of automated tools such as computer-aided design (CAD), Unified Modeling Language (UML), and modeling, simulation and analysis" [133]. While it has previously been recognized that more analytical support in Pre-milestone A is needed, CAD is typically used primarily in preliminary and detailed design, which occur post-Milestone A. Furthermore, the use of CAD generally implies a materiel solution, when pre-Milestone A should, according to the DoD 5000, consider more than just materiel solutions. Thus, while this criteria may sometimes be appropriate, it is unclear at what times it should be applied, and certainly should not be a blanket criteria in pre-Milestone A. Many other criteria like this can be found in the DAPS Methodology documentation. In addition, some of the criteria are vague and it is unclear what the standard would be for complying with these criteria. For example, again taken from 'Systems and Software Engineering (SSE) Tools' Pre-Milestone A, one of the criteria is "Engineering analysis and designs are supported by appropriate diagramming and design tools" [133]. Another example can be found in the Factor 'Architecture', under the Pre-Milestone A and Pre-Milestone B criteria. One of the criteria is "The open architectures employed in the system should satisfy the specified performance and support requirements" [133]. In both of these cases, there is no clear standard for determining whether the criteria have been met, thus creating a high level of ambiguity in execution of the methodology. A similar set of criticisms can be applied to the focus areas.

In summary, DAPS provides a list of criteria and relevant questions for program managers and decision-makers to consider during the acquisition process, but does not provide adequate guidance as to how the necessary information should be gathered or the standards for quality. Furthermore, it focuses more on the managerial aspects of moving through the acquisition process and less on the analysis behind it. Overall, while it has some guidance for a systems engineering methodology used in the acquisition process, it is not in and of itself a systems engineering methodology, and thus is not an acquisition support methodology as described in the context of this thesis. However, it should be noted that, in general, the systems engineering criteria specified here for Pre-Milestone A activities include many of the criteria discussed previously, including well managed requirements that are traced between product design and operational capabilities, integrated tools and data, an easily up-datable framework, use of quantitative analyses, and performing analyses early when possible to avoid costly changes later in an acquisition program.

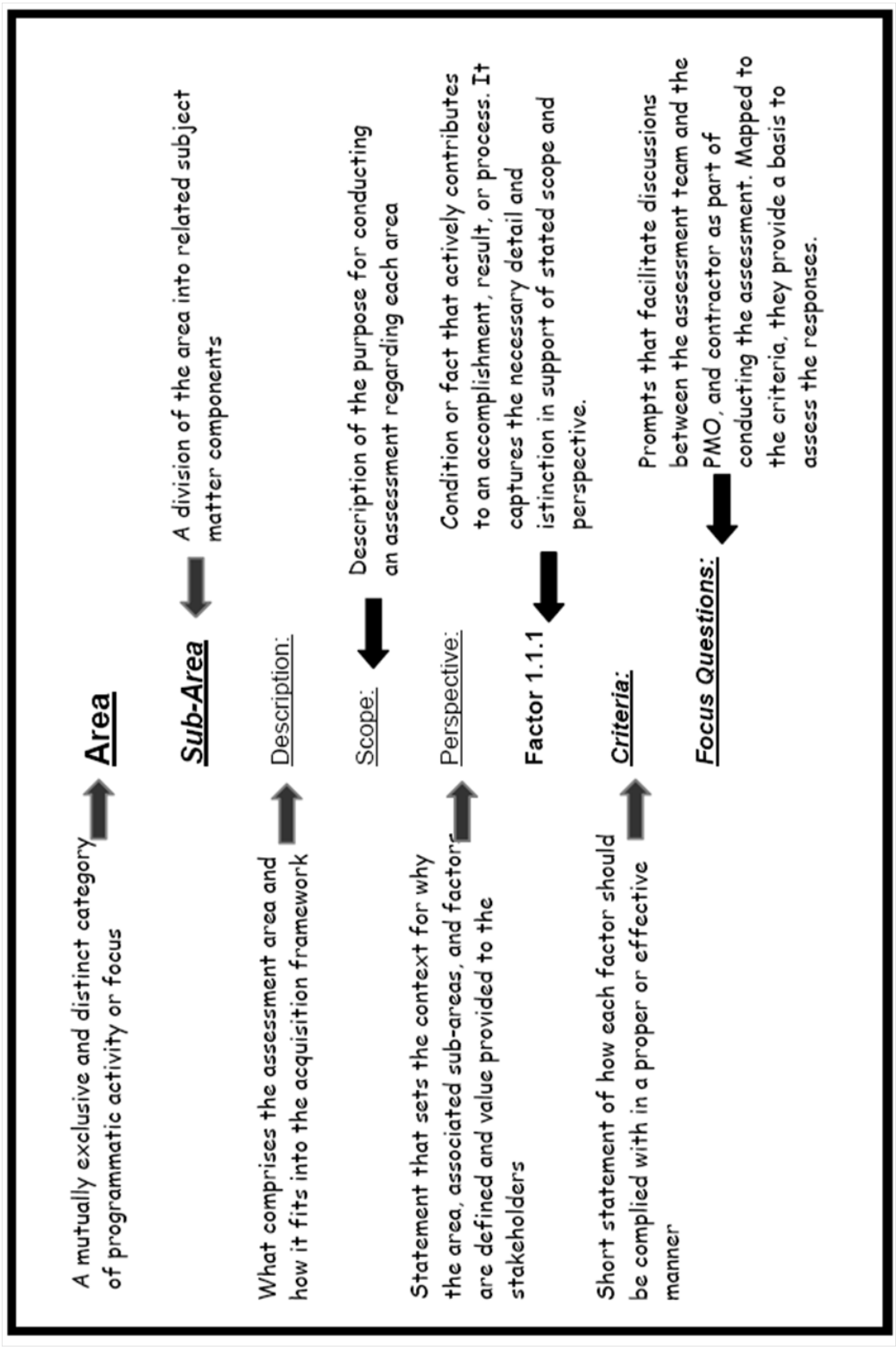


Figure 9: DAPS Methodology Overview. Reproduced from [133]

2.2.2 Air Force CBA Methodology

The United States Air Force has specified guidance for what it refers to as 'Early SE', which is documented in the United State Air Force Early Systems Engineering Guidebook [167]. According to this guidebook, early SE has four segments, the CBA, the Concept Exploration and Refinement (CER), the Preferred System Concept (PSC) maturation, and the Technology Development (TD). The recommended process for the CBA and CER is depicted in Figure 10. According to the Air Force's guidance, both the CBA and CER are conducted as part of the pre-Milestone A analysis. This guidance explicitly calls out the need for an SoS perspective and the use of architecture in early phase analysis. A brief summary of the guidance will be provided here, as well as some discussion of its advantages and disadvantages. However, it is important to note that what is presented here is guidance for best practices, and not a methodology. Like the CBA manual itself, the guidance focuses more on *what* should be done than *how* it should be done.

The process begins when authorization to proceed with a CBA is granted. This includes the identification of a capability focus area with clear capability requirements and guidance. This leads into the Capability Documentation and Analysis, which begins with a list of needs or shortfalls which are then further decomposed and focused into 'quantifiable tradespace boundaries'. In fact, the guidebook states that [167]:

> The most important part of the process is taking the initial input requirement and decomposing it into quantifiable tradespace boundaries. The broader the tradespace, the longer the process will take; in contrast, if the tradespace is limited too far, it will yield a single point design. The balance between these extremes is based on the time, effort, and resources dedicated to a particular iteration.

Although the phrase 'quantifiable tradespace boundaries' is not explicitly defined, the

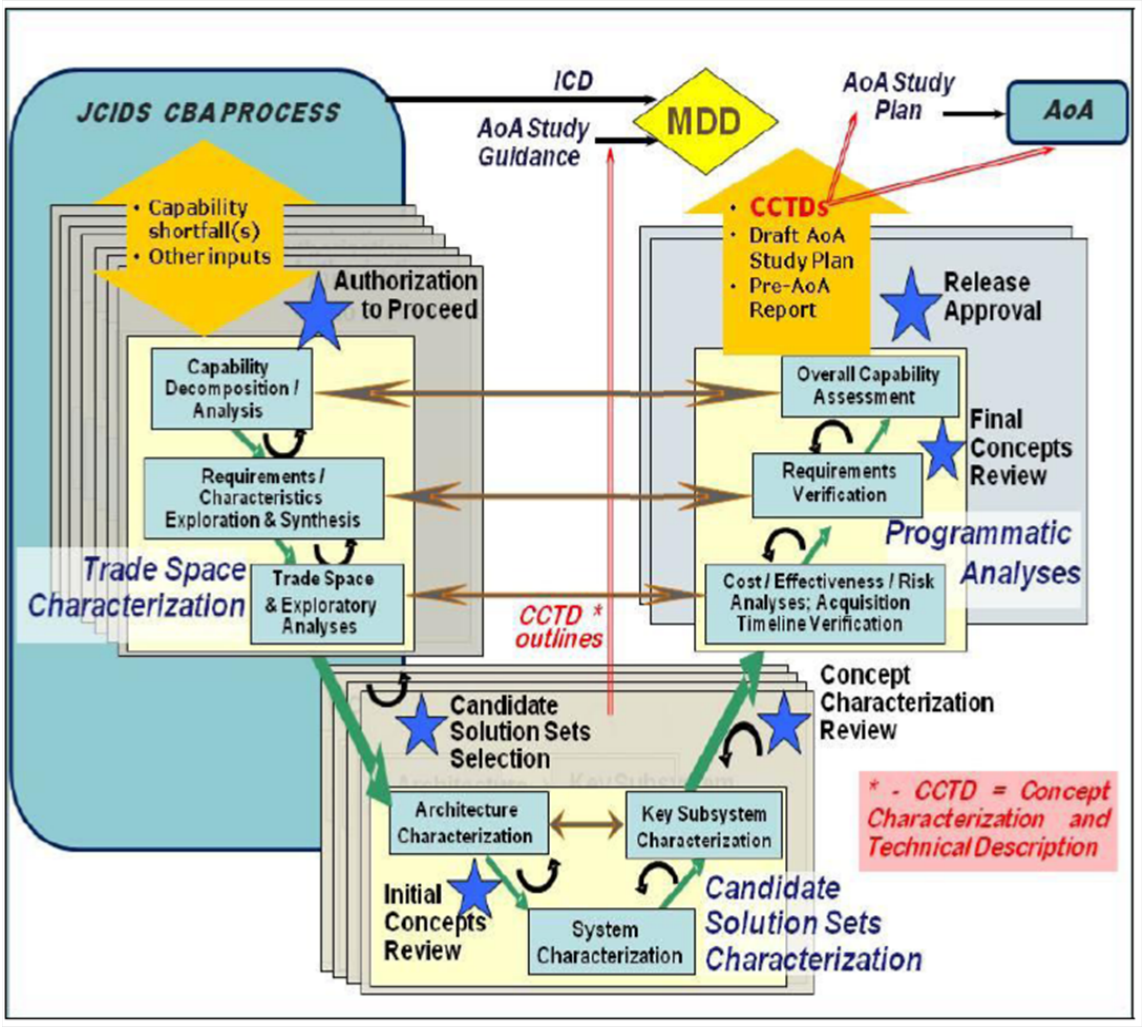


Figure 10: Air Force Early Systems Engineering Process. Reproduced from [167]

following example is provided as clarification on the term [167]:

> Stated mission task: *Provide the capability to Find/Search, Fix, Track, and Characterize all man-made space objects, space events (space launches, maneuvers, breakups, dockings, separations, reentries and decays) and space links (ground to space, space to space, space to ground) for near-Earth and deep space orbits.*
>
> - Decomposed Concept Engineering mission task: *Provide the capability to Find/Search, Fix, Track, and Characterize all man-made space objects in Geo-stationary/Geo-synchronous orbits.*

In addition to defining the tradespace boundaries, a requirements baseline is established using an Initial Capabilities Document (ICD), Concept of Operations (CONOPS), or Capstone Requirements Document. However, since this step is performed during pre-Milestone A, it is unlikely that an ICD would be available. The guidance then suggests that existing systems and capabilities be compared to these requirements to corroborate the stated shortfalls and find indications of any available technologies to close the gaps [167].

The next step of the process is to do Requirements/Characteristics Exploration and Synthesis. In this stage of the process, users need to being to map the quantified requirements from the previous step to potential materiel and non-materiel solutions. It is recommended that these alternatives be described with an OV-1 Operational Concept Graphic combined with an Operational Concept Narrative. This should include the documentation of anticipated trades and the rationale for decision-making based on those trades. In order to collect potential alternatives, use of previously identified concepts (from previous studies) is recommended to be combined with industry days and group brainstorming sessions [167]. It should be noted that while this

approach will result in the consideration of multiple alternatives for consideration, it is not systematic, does not guarantee that the full design space will be considered, and may be affected by biases from those included in the brainstorming. Furthermore, this process is far from repeatable, as multiple attempts at the process could result in completely different concept sets.

Once the concepts are identified, the Trade Space Exploratory Analysis begins. In this stage, the high level concept alternatives are compared against one another and against the previously identified shortfalls. This should, according to the guidebook, include estimations of each concept's feasibility, and concepts should be ruled out if they are perceived to violate physical laws or based on 'engineering rules of thumb'. If a concept is eliminated, the reason for elimination should be included in the documentation. After this is done, the guidance recommends that solutions be sorted by their maturity, so that those with a low TRL or those that are judged to be mid-term (9-15 years) and far-term (15-23 years) solutions should be ruled out or accompanied by a technology maturation strategy [167]. The challenge with this recommendation is that a large number of alternatives are either never considered or are weeded out with little concrete justification when they are still only partially conceptually formed. This can result in cognitive biases due to Inference of Impossibility and Denying Value Tradeoffs, as discussed in [155] and summarized in 2.1.1.

The formulation of metrics is also done in the Trade Space Exploratory Analysis step, and first-order models or intelligent assessments are recommended to be used to eliminate those alternatives which are not sufficient according to assessments of these metrics [167]. It is concerning that the metrics are not formulated until this step, as the Capability Documentation and Analysis step called for quantified requirements. This suggests that the measures against which solutions are evaluated are not the same metrics that were used to describe requirements, leading to the possibility of a mismatch between recommended solutions and capability-level needs. It is also noted

that the use of first-order modeling early in the evaluation of alternatives process to downselect the alternatives space is likely preferable to intelligent estimates because the estimates may by subject to the cognitive biases as discussed in 2.1.1.

The next step of the process is the Candidate Solution Set Selection. This step uses the information collected in the preceding steps to downselect to a small number of candidate solutions based on technology maturity, expected fielding time frame, satisfaction of needs, the absence of similar efforts in the military community, and resource availability. This concludes the Tradespace Characterization Phase of the process [167].

The next three steps of the process make up the Candidate Solution Sets Characterization Phase, and these three steps are designed to reflect traditional systems engineering. There is an implicit assumption that all remaining alternatives at this phase are materiel (system) alternatives, and this is evident in the steps included in this methodology, as well in the application of traditional systems engineering at this stage in the process. This phase begins by performing Architecture Characterization. In this step, DoDAF views beyond the original OV-1 are created, including the OV-2, OV-3, OV-4, and OV-5. This is done so that initial estimations on interface requirements can be made and integration needs can be understood. Solutions are then evaluated based on their ability to meet interface and integration needs. At this point, it is suggested that modeling and simulation using a wargame may be possible in order to evaluate concepts using a representative scenario [167]. There are several issues with the assumptions of this stage. First, it assumes that all remaining concepts will be able to be evaluated in the context of a single operational architecture, implying that only one operational alternative concept can remain at this point. Second, it assumes that knowledge of the operational architecture products is sufficient for assessing interface and integration needs and conducting wargame-level simulations. However, without documenting what other systems are being assumed

in the SoS, it will be very difficult to assess a new system's ability to interface with the other systems or to assess the performance of the SoS in a wargame or other simulation. It should be noted, however, that this stage represents a second level of modeling and simulation for further downselection.

Once the Architecture Characterization is complete, an Initial Concept Review begins. At this stage, each remaining concept is reviewed and a decision is made to carry this forward to the next step, to shelve the concept, or to return to the previous steps to further refine or modify the concept. Those concepts that are selected to be carried forward are then taken to the next stage of the process, which is the System Characterization. At this stage, systems are developed to a further level of fidelity, and key system architecture products are developed, including the SV-1, SV-3, and SV-4. This stage also begins the process of developing a technical requirements document to specify the justification for design decisions, system configurations, and trade studies. Subsystem Characterization is done as a sub step to make sure that the immature technologies being used as subsystems have an acceptable technology maturation plan. This results in the creation of the SV-7 and the SV-9. At the end of this phase, all the candidate system designs are reviewed a subset are selected to be carried into the next phase of the process [167].

The Programmatic Analysis Phase focuses on ensuring that the remaining concepts are viable in terms of resource, schedule, and cost constraints. This is done by first performing a Cost/Effectiveness/Risk Analysis and Acquisition Timeline Verification step, then performing a Requirements Verification step. At the end of this phase there is an final concepts review which results in a set of closed concepts to be carried forward into the next phase of analysis. Documentation is produced and this closes the CER phase of the systems engineering process [167].

After this point in the process, the documentation goes on to state that a detailed

AoA is performed on remaining concepts to downselect to a single preferred alternative and the PSC and TD stages are begun to lead up to the Milestone A decision. However, the details for these stages of the process are not defined, and are instead earmarked for inclusion in a future revision. As such, the guidance is incomplete and unable to take a user fully through the process to pre-Milestone A [167].

As was noted previously, there is not a full methodology contained within this guidance, but rather a process with information about what should be done (for the first half of the process) and not how it should be done. For example, although a multi-fidelity modeling and simulation approach with several downselections is recommended, there is little to no guidance on what types of models are appropriate for use. Furthermore, is unclear that a full gap analysis is required as part of this process. Metrics derivation is done very late in the process, after some concepts have already been eliminated from consideration. It is also observed that the overall process is biased toward materiel alternatives, as it is assumed that after the first phase of the process only system alternatives remain. While the approach does consider multiple alternatives, downselection to a single alternative is done halfway through the process (between CER and PSC). For each individual step, shortcomings of the steps have been discussed in conjunction with that step. Overall, insufficient guidance is provided to fully execute a CBA.

2.3 Research Questions

Since it has been determined that development of new methodology for acquisition decision-making during CBA is needed, and further recognizing that in general, acquisition decision-making processes have the steps of problem formulation, gap analysis, alternative generation, alternative evaluation, and alternative selection, several research questions are inspired in each area, and are discussed in this section. It is

assumed that problem formulation is actually an input to this process, since a potentially new acquisition comes about when an initial potential problem area has already been identified. Thus, it is assumed that at least an initial description of the hypothesized problem will be available as a starting point for this methodology. However, in order to create the methodology, a set of criteria that are desirable for the methodology to possess are needed. This will allow the appropriateness of the methodology to be assessed, and will also be used to help guide the selections made between the tools and techniques which support the methodology. This spurs an initial research question on the criteria with which the methodology should be assessed, which will provide guidance in answering other research questions, and thus has been dubbed Research Question 0, and is shown in Table 4.

In order to develop a methodology, a generalized framework for the overall process is required. Since existing systems engineering methods purport to include many of these same steps for the development of a single system, and since it was determined earlier that systems engineering principles should be leveraged in the development of this process, these methods may provide a good starting point for developing the process structure. Once the process structure is determined, the known steps will need to be integrated into the framework, and then it will need to be determined if there are additional steps required to ensure information flow between the existing steps. For example, given that several of these steps will require evaluation of the baseline or alternatives against a set of metrics, it is noted that a step to derive a set of metrics for the analysis is required as well. The research questions that need to be answered in the development of the methodology with respect to the overall model of the method are shown in Table 4, under Research Question 1.

Once the overall steps and structure of the process are in place, the next task is to explore how each of these steps should be accomplished in the context of CBA and acquisition decision-making. Thus, Research Questions 2-6 (shown also in Table 4)

explore how each of these steps should be performed. Recognizing that many of the steps will require a set of metrics as an input and/or output, metrics derivation is an important step in the methodology. This inspires two research areas. Determining a repeatable and structured way to develop metrics from a top-down perspective starting with potentially vague capability requirements and ending with clear, measurable objectives is required. However, it is important that the resulting metrics clearly and fully demonstrate fulfillment of capability requirements, and that these metrics can be obtained in a cost-effective and timely manner. As a result, it is important to not only determine a list of candidate metrics, but also to examine the goodness of these metrics in the given application. This inspires Research Question 2, regarding first how to derive the metrics and secondly how to verify their goodness.

Research Question 3 addresses the area of gap analysis. There are two aspects to be considered in gap analysis. First, an approach for performing gap analysis must be identified. However, once gaps are identified, they need to be quantified and ranked. Thus, a second necessary research element in the area of gap analysis is an approach to ranking the gaps and developing a prioritized list of gaps.

Once gaps are identified and prioritized, candidate alternatives for filling these gaps must be identified. This is not a trivial prospect. The large and diverse space of SoS alternatives within the DOTMLPF spectrum provides unique challenges to alternative identification. Simply managing the size of the alternative space and ensuring that alternatives of all types are put on the table is a challenge in itself. In addition, recognizing that alternatives may result in changes that need to be propagated throughout the architecture, it is necessary to determine how to limit the alternative space to only those alternatives that are architecturally feasible so that precious resources are not wasted evaluating alternatives that could never actually be implemented. This inspires Research Question 4.

Once the alternatives have been identified, a way of evaluating and comparing

these alternatives is required to prune down the alternative space to a manageable number for decision-making. There are many existing qualitative and quantitative techniques for evaluating architecture alternatives. Since it is unlikely that any one tool will be able to provide the full set of needed evaluations, it is likely that an set of modeling tools will be required to gain as much insight as possible into all metrics. In addition, limited information is available in the early phases of acquisition, and modeling tools must be selected that are able to perform analysis using this limited set of information. The process also needs to recommend what minimum set of information should be made available to enable different types and fidelities of modeling. Since the baseline architectures will likely be documented using DoDAF, it is of interest to explore how DoDAF can be made executable and links between the architecture specification and modeling and simulation can be developed. However, it is likely that the information provided in the DODAF products will need to be supplemented to enable modeling, and this will need to be explored as well. These considerations inspire Research Question 5.

Finally, the information obtained in the evaluation of alternatives needs to be presented to support decision-making. This environment will need to integrate information from multiple models and include the ability to display the architectures of alternatives. It will also need to show how the performance of alternatives map back to capability needs, and be able to compare alternatives based on their performance at the capability level. Identification of needs and implementation guidance for a decision-support framework is addressed is Research Question 6.

Table 4: Summary of Research Questions

Number	Question
(0)	What are the criteria for a successful CBA methodology?
(1)	(1.1) Which model of the systems engineering process, if any, is appropriate as a basis for systems engineering in the CBA decision-making process and why?
	(1.2) How should the steps of the methodology be arranged within the chosen model?
	(1.3) What are the inputs and outputs of each step, and are additional steps required to support linkages between these steps in the model?
(2)	(2.1) What is a repeatable and structured way to derive a set of metrics from a top-down capability-based perspective that adequately measures baseline and alternative performance, cost, and risk?
	(2.2) How should these metrics be tested for goodness?
(3)	(3.1) What approach is best suited to robustly identify gaps in current defense SoS?
	(3.2) How should the gaps be ranked in order to capture their size, criticality to mission objectives, and uncertainty in the estimations?
(4)	(4.1) What is a reliable way to develop the large and diverse set of DOTMLPF alternatives for an SoS?
	(4.2) How can the alternative space be reduced to include only the subset of alternatives that is architecturally feasible?
(5)	(5.1) How can quantifiable and traceable analysis of the generated architectural alternatives be performed in a timely and cost effective manner for the architecturally diverse set of alternatives considered in the early phases of defense acquisition?
	(5.2) Are the architecture descriptions provided by DoDAF products sufficient models for answering early-phase acquisition questions and what limitations does DoDAF have for executable architecting and how can these issues be resolved?
	(5.3) What modeling tools can be used in the early phases of acquisition to provide insight into metrics of interest while using only the limited amount of information available early in the process?
	(5.4) How should information from various modeling tools be integrated to support analysis of gap fulfillment?
	(5.5) What subset of specified DoDAF products combined with a set of fit-for-use products will provide the information necessary for early-phase, acquisition-focused executable environment? How does this depend on the selection of modeling and simulation tools?
(6)	How should a decision-support environment to improve early-phase acquisition decision-making be developed, and what should be the features of such an environment?

CHAPTER III

BACKGROUND

This chapter contains background information on existing techniques found in the literature that are candidates to address the research questions presented in the previous chapter. Each section of this chapter corresponds to one of the groups of research questions, addressing first the overall structure, and then each step of the overall methodology. The intent of this chapter is not to provide a comprehensive discussion of each technique, but rather to give enough background on each of the techniques to an unfamiliar reader to enable a comparison of the techniques during the methodology development in the following chapter.

3.1 Existing Systems Engineering Models

There are several common standard systems and software engineering process models: the vee model, the waterfall model, and the spiral model. All three of these models originated in the field of software development, and were aimed at improving the management of the development of large software systems. As software products became larger, more complex, and more modular, these models were proposed to improve the way software development was performed to reduce cost and risk and minimize rework and design changes late in development. The waterfall and the spiral model have been most commonly applied in the field of software engineering, while the vee model has been primarily applied in the field of systems engineering. While many other models for systems engineering have been proposed, these three models have provided the backbone for many model-based systems engineering methods (MBSE) which encourage the use of models throughout the design and development process.

This model-based approach is a paradigm shift from the previously used document-based approaches [64]. NASA also has extensive engineering guidance for managing complex systems development and deployment. The NASA SE Engine will also be discussed, along with its relation to NASA's acquisition decision making. Many other established SE models and practices exist, such as the process documented in the Naval Systems Engineering Guide [171], but as these processes largely pull from the methods being discussed here, they will not each be discussed in detail in this thesis document.

3.1.1 The Waterfall Model

The waterfall model was proposed in 1970 by Winston Royce [151], in response to his observation that the two traditional software development steps, analysis and coding, were not sufficient for the development of large software systems, but that developers and customers were unwilling to invest in additional planning and testing steps because they increased cost and did not directly contribute to the final product [151]. The waterfall model developed by Royce is specifically geared toward the design and development of large software packages, and is depicted in Figure 4 [151]. Royce's waterfall model added steps to the traditional software development paradigm for requirements analysis, program design, and testing. The basis for the model is a purely linear waterfall, but he recognized quickly that some iteration with previous steps in the process was unavoidable. His goal was to keep this iteration between successive steps (as opposed to non-successive steps) as much as possible, recognizing that this would scope the required changes and minimize costs associated with change efforts. He noted, however, that in reality this was not always possible. He noted two primary cases where iteration would occur with steps far upstream. The first is when issues are discovered in testing that require a major redesign of the software, and the second is when issues are discovered during the design that require requirements to

be rethought. This is depicted to the left of the dashed line in Figure 11.

Royce also suggested five additional features that must be added to the basic approach in order to mitigate risk, and these features are depicted to the right of the dashed line. The five risk-mitigating features are [151]:

1. Program design comes first. This implements a preliminary design stage using software designers prior to performing analysis.

2. Document the design. This involves documentation at all phases of the development process to help manage requirements, track decisions, manage interfaces, and implement testing.

3. Do it twice. This involves the use of a prototype or simulation prior to development of the final product to identify critical problem areas.

4. Plan, control and monitor testing. This involves rigorous management of the testing phase to manage cost, schedule and risk, and ensure testing produces accurate results.

5. Involve the customer. This means including the customer in a series of reviews prior to the final delivery of the product to be sure that customer has committed to the actual product that is being delivered.

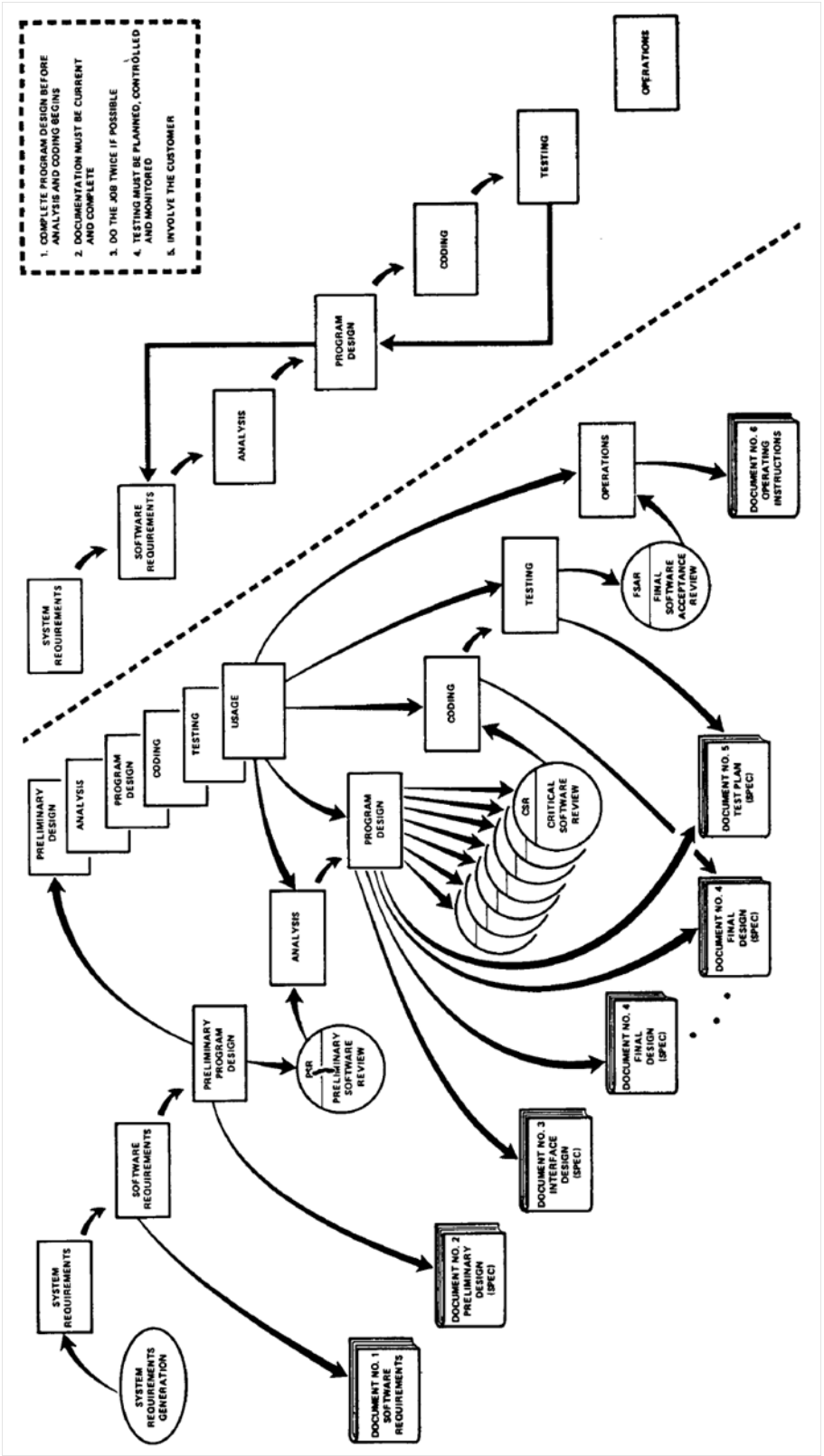


Figure 11: Original Waterfall Model. Reproduced from [151]

3.1.2 The Spiral Model

The spiral model of design was first proposed by J. Harvey Evans in the late 1950s. He proposed the spiral model as a solution to ship design, claiming that for complex problems (such as the design of a ship or aircraft) "which necessarily involve more compromises owing to the increased state of incompatibility resulting from the many more requirements, these initial estimates and decisions may be critical" [65]. His spiral model, proposed for the design of a cargo ship, is shown in Figure 12 [65]. The general philosophy behind the model is that design tasks can be organized in a logical way such that successive iterations will rapidly converge on the "ultimate, refined, and balanced solution" [65].

In 1988, Boehm recommended a spiral model for software development as an improvement to the waterfall model [20]. Boehm believed that the waterfall model, which had become the standard for software acquisition by 1988, was appropriate for compilers and secure operating systems, but failed in the case of interactive, end-user applications due to excessive documentation of poorly understood interfaces that resulted in the generation of large amounts of unusable code, and therefore overruns in schedule and budget. He also cited advancements in programming languages, saying that in fourth generation languages, elaborate documentation prior to implementation was unnecessary. Unlike Evan's spiral development model, where successive iterations are conducted as the spiral converges to the center, Boehm's model represents successive iterations as moving outward around the spiral, with the radial dimension representing cost. Each iteration of the spiral begins with identification of objectives, alternatives, and constraints. Next, alternatives are evaluated and sources of risk are identified. Mitigation strategies to reduce risk are developed, which could include anything from benchmarking to simulations to prototype development. Each cycle of the spiral concludes with a review. The spiral model has the advantages of using a risk-mitigating approach focused on the life cycle of the software and an early AoA

to identify cases where reuse or redesign of previous software are viable options. In addition, it has been shown to be useful for the simultaneous design of hardware and software, which is a common practice in the computer industry [20]. Boehm's spiral model can be seen in Figure 13.

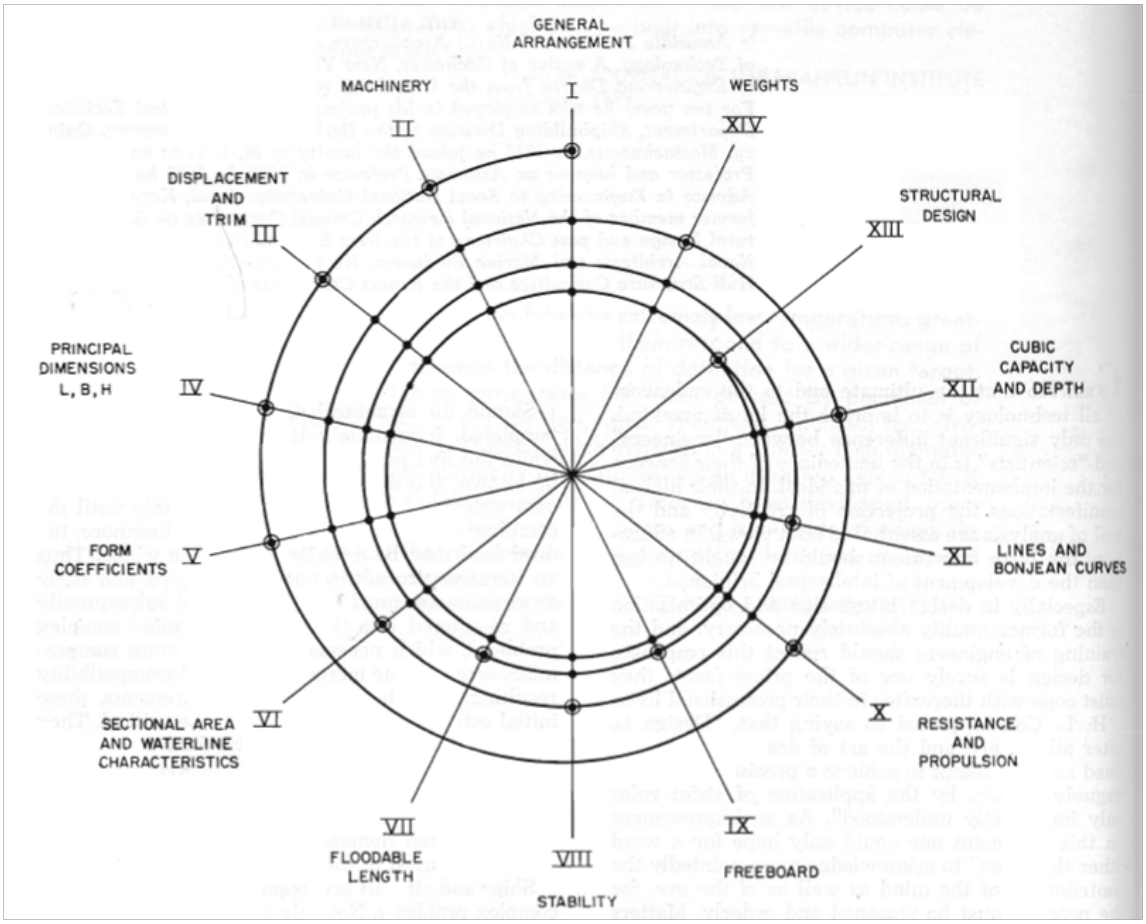


Figure 12: Evan's Spiral Design Model for Ship Design. Reproduced from [65]

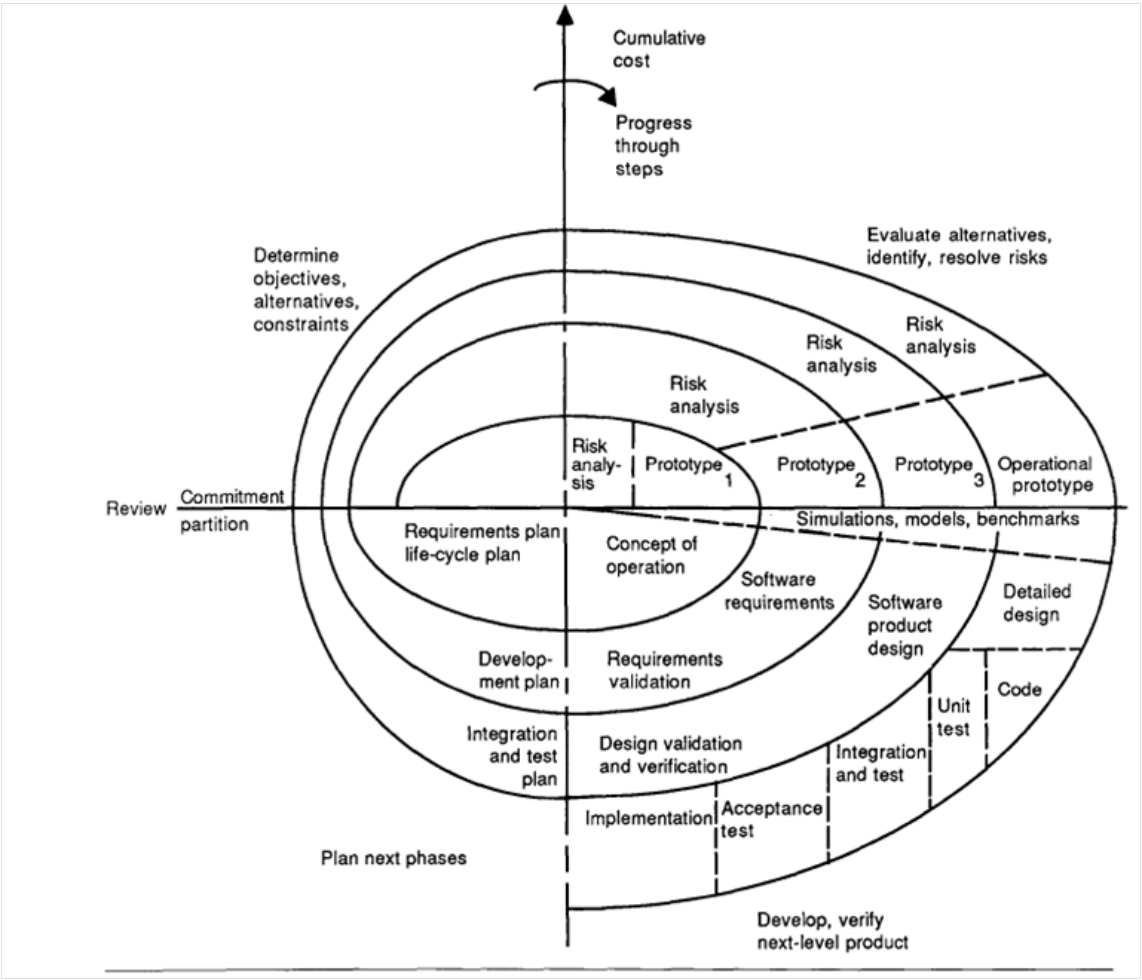


Figure 13: Boehm's Spiral Model for Software Development. Reproduced from [20]

3.1.3 The Vee Model

The vee model was originally developed in parallel by NASA and by Forsberg and Mooz. NASA's vee model was developed in 1987 as a software development method. [125] In 1990, Forsburg and Mooz [67] presented the vee+ model for systems engineering of hardware and software systems, which is shown in Figure 14. However, for the purpose of this work, and according to common convention, Forsberg and Mooz's vee+ model will be referred to as the vee model, and the vee model developed by NASA will be referred to as the NASA vee model.

NASA's vee model was developed and adapted in NASA's Software Management

and Assurance Program in 1987. It was derived from the waterfall model, adding the idea of multi-level decomposition and integration to move to a vee shape. The depth of the vee reflected the level of complexity of the software system, as more complex systems required more detailed decomposition and integration. It also added a third dimension, which was depicted as normal to the page, to reflect the timeline of incremental or multiple deliveries [125] However, more recent NASA guidance on software assurance has abandoned this vee model, and NASA has moved to the NASA Systems Engineering Engine as a Systems Engineering model, which will be discussed in more detail in the following section [128, 129].

Forsberg and Mooz's vee is comprised of a definition and decomposition phase (traversing down the left side of the vee), and implementation phase (across the bottom) and an integration and verification phase (traversing up the right side of the vee). The left side of the vee follows the waterfall model established by Evans. Unlike the waterfall model however, the vee does not prohibit doing more detailed work earlier in the process to establish or show feasibility of alternatives, or clarify requirements. In fact, the use of concurrent engineering is strongly recommended. The vee process attempts to minimize backward iteration once formal decisions have been made. These formal decisions are made at a series of control gates. Integration and verification is performed as the process moves up the right side of the vee. These activities directly correspond to the activities on the left side of the chart. As requirements are developed on the left side, corresponding verification requirements should be developed for the right side simultaneously. These requirements ensure that the developed product meets the intentions of the designers [67].

Forsberg and Mooz have proposed several improvements to their original vee model. For the purpose of this work, the most notable improvement is the architecture and entity vee models, which are then combined to create the dual vee model. The architecture vee is aiming at managing the overall architecture of a system that is

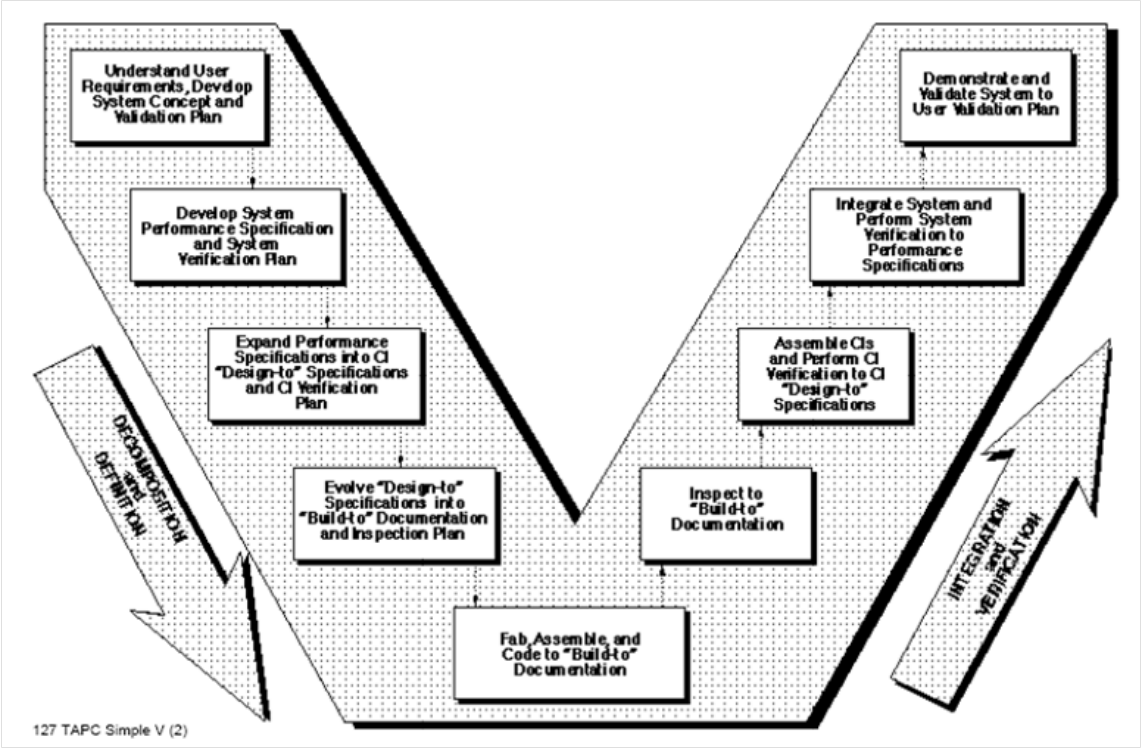


Figure 14: Forsburg and Mooz's Vee Model. Reproduced from [67]

being developed and fielded. The architecture vee provides the 'what, why, and who' for each level of the system architecture. The architecture vee is depicted in Figure 15. This model recognizes that design and development are iterative, but overall assumes a left to right time flow, meaning that the architecture decomposition and definition is done at increasing levels of granularity down to the lowest configuration item, and then integration and verification are performed at decreasing levels of granularity [66].

At each level of the architecture vee, one or more entity vees are required to describe the process for obtaining each element of the architecture. This could include the element being developed, being purchased, or obtaining this element by any other means. It is necessary to have an entity vee for every single element of the architecture, at each level of the architecture. Because of this requirement, it is possible that a large number of entity vees are required for any architecture development. For each element, the entity vee describes the process of defining the requirements for the

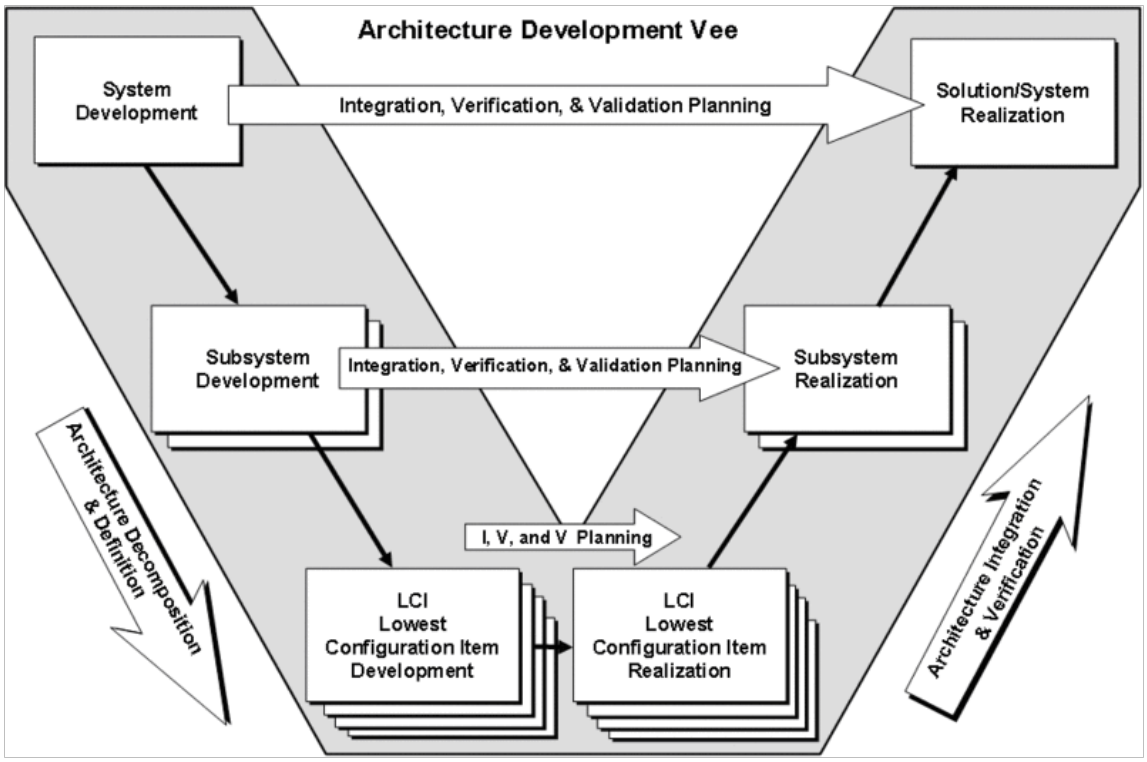


Figure 15: Forsburg and Mooz's Architecture Vee Model. Reproduced from [66]

element, determining the source of the element, designing and building the element (if necessary), verifying that the element will perform as expected, and validating that the element meets the original requirements for that element [66].

The architecture vee and the entity vees at each level are then combined to create what Forsberg and Mooz refer to as the dual vee. The dual vee is pictured in Figure 17. The full realization of the architecture occurs through the process of moving through the architecture vee and the entity vees simultaneously to ensure that all of the elements required to develop the architecture are acquired and tested in isolation, and that they are integrated and tested in the context of the overall architecture performance, and together meet the requirements of the integrated systems. The dual vee model also allows key design reviews for elements and for the architecture to be coordinated in a phased sequence that allows decisions to made in a logical order at each level of the architecture [66].

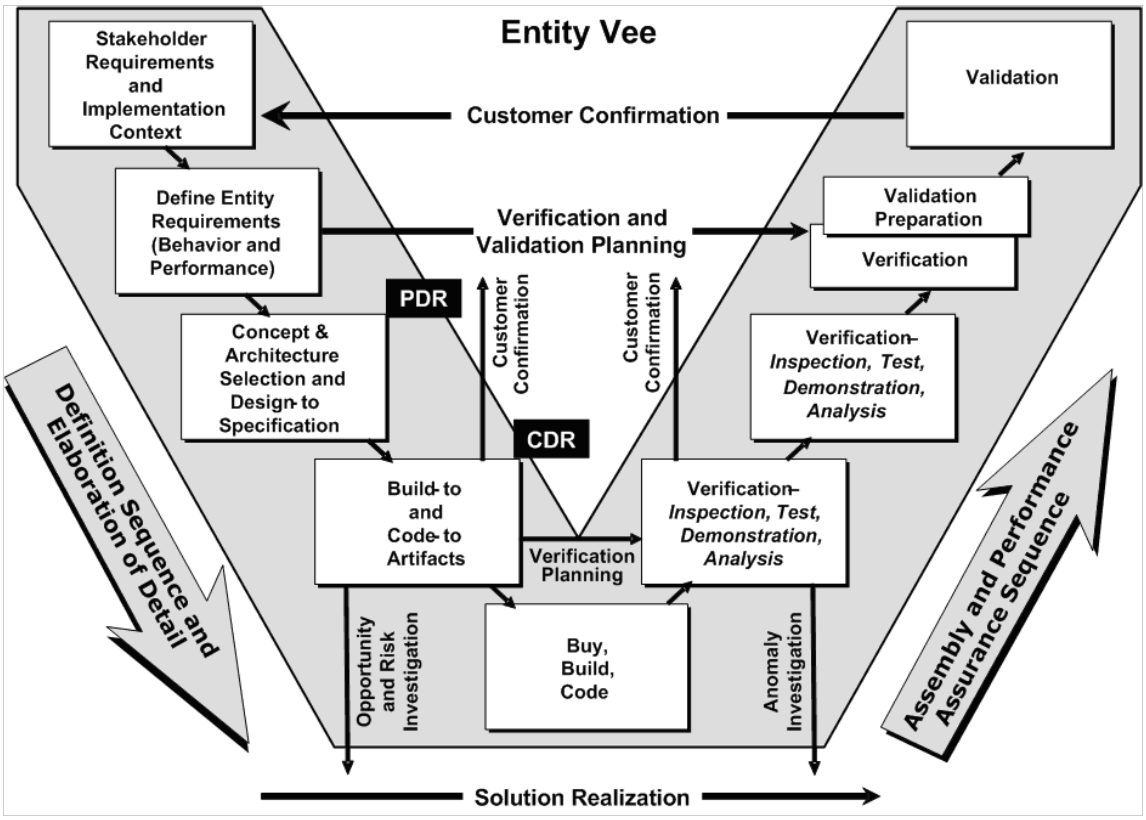


Figure 16: Forsburg and Mooz's Entity Vee Model. Reproduced from [66]

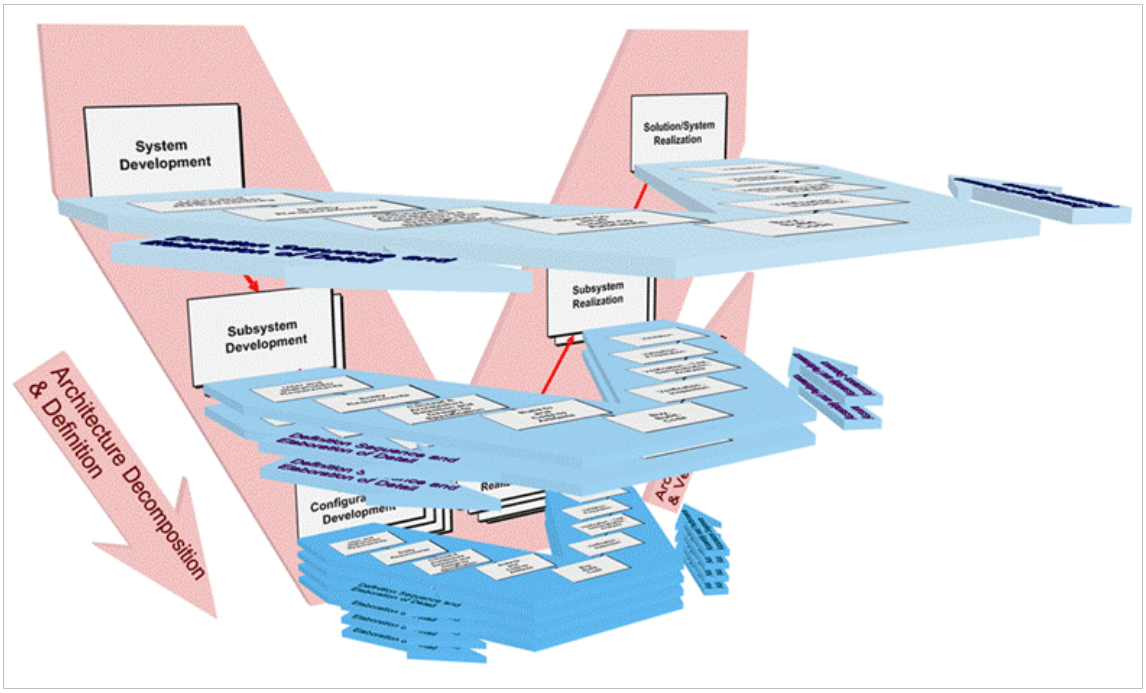


Figure 17: Forsburg and Mooz's Dual Vee Model. Reproduced from [66]

3.1.4 The NASA Systems Engineering Engine

NASA has a required systems engineering process that is specified in NPR 7123.1A [127]. This standard is accompanied by the NASA Systems Engineering Handbook [129]. The NASA SE standard framework is comprised of three main elements. The Common Technical Processes deals with design, realization, and technical management. The Tools and Methods includes common terminology, a view of a system, documentation, life cycle models and views, and management policies, procedures and practices, and resources. Finally the Workforce includes the skills, competencies, teamwork, ethics, and training. Put another way, the Common Technical Processes deal with the engineering of the system, the Tools and Methods deal with the enablers for efficiently carrying out the engineering, and the Workforce element deals with the human resources and expertise required. Together, these three elements provide SE capability to NASA. This is summarized in Figure 18 [127].

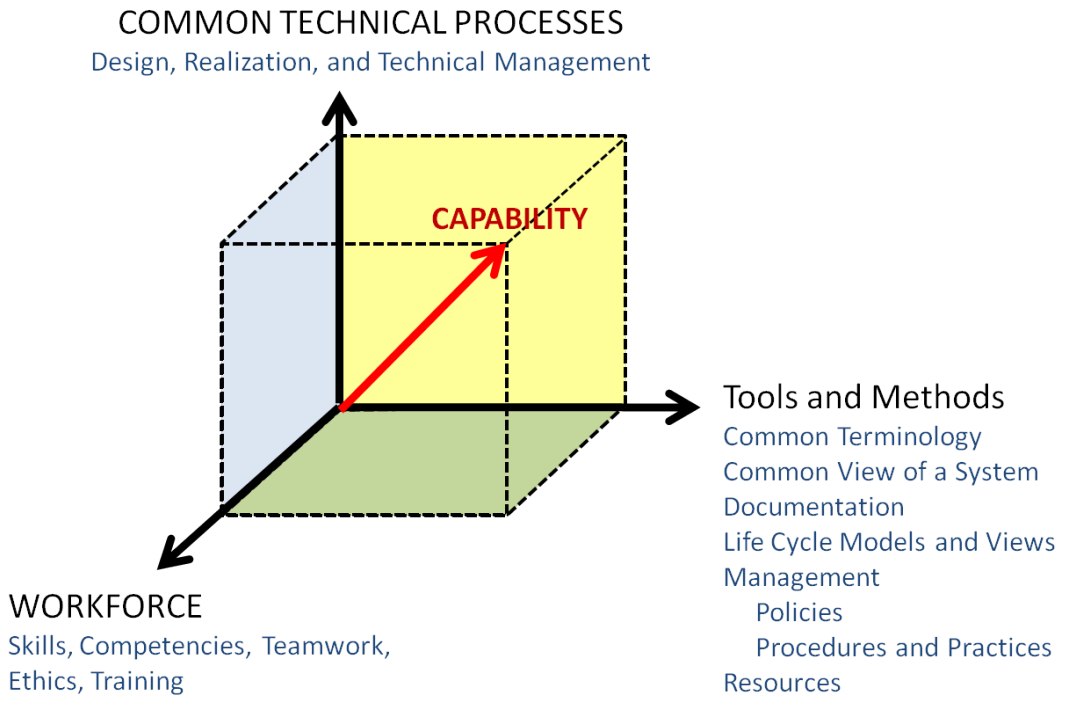


Figure 18: NASA's Systems Engineering Framework. Recreated from [127]

The NASA SE process also specifies the use of the NASA SE Engine to guide technical processes and requirements for SE throughout the life-cycle. The NASA SE Engine is depicted in Figure 19 [127]. The NASA SE Handbook provides guidance on the use of this engine for SE. The first column of this diagram deals with requirements definition and the technical approach, so this column is the most relevant to CBA. The other columns discuss the technical management process of overseeing a new design and the building and testing of the new design, which would happen post-Milestone A in the DoD. As such, only the relevant pieces of this model to CBA will be discussed here. More information on the NASA SE Process can be found in [127, 129].

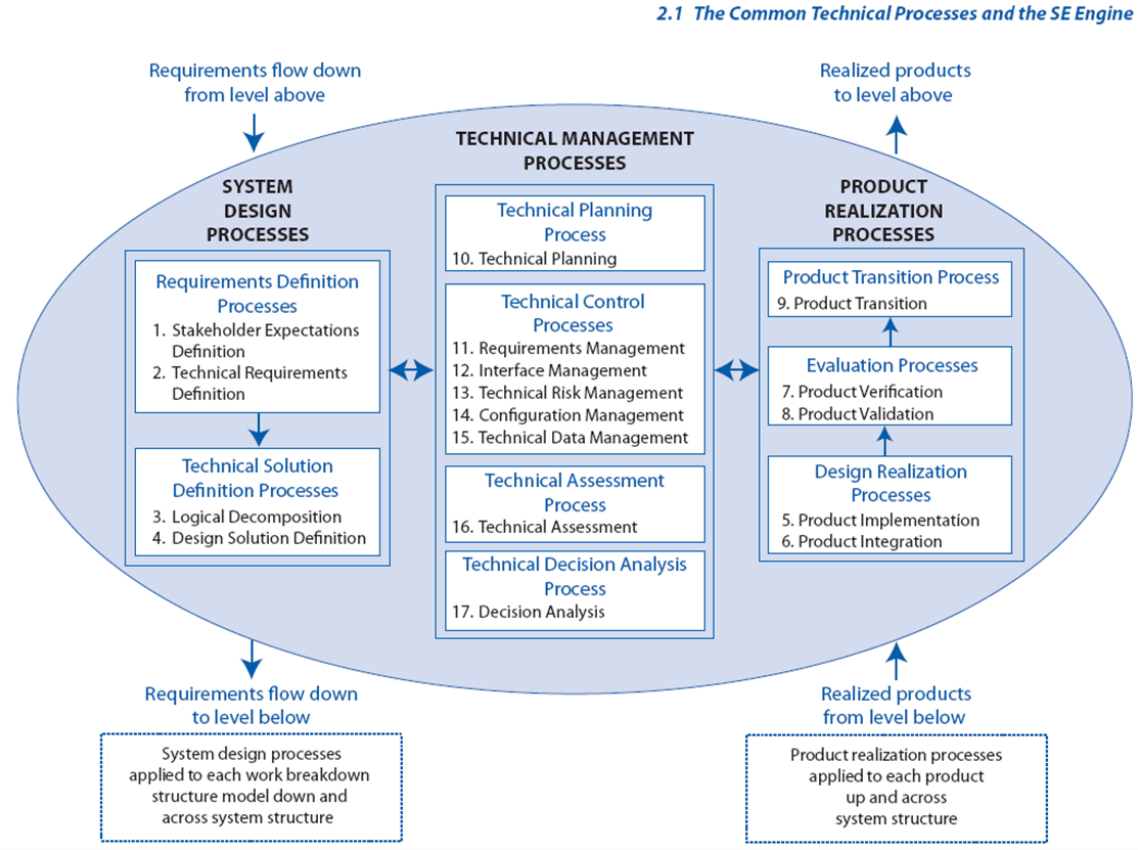


Figure 19: NASA's Systems Engineering Engine. Reproduced from [129][127]

The system design processes shown in the left-hand column of the NASA SE

Engine are top-down processes geared towards clarifying the requirements of stakeholders and transforming these requirements into executable technical requirements that are then used to create a defined concept to carry forward to the design phase. For a complex or complicated system with many elements, all of the processes are repeated hierarchically until all components are at the level at which they can be bought, reused, or designed. Like the DoD, NASA has created an overall process flow of the project life-cycle, beginning with what is called Pre-Phase A: Concept Studies, all the way through Phase F: Closeout. The SE Engine works within this process flow, of which a simplified version is shown in Figure 20.

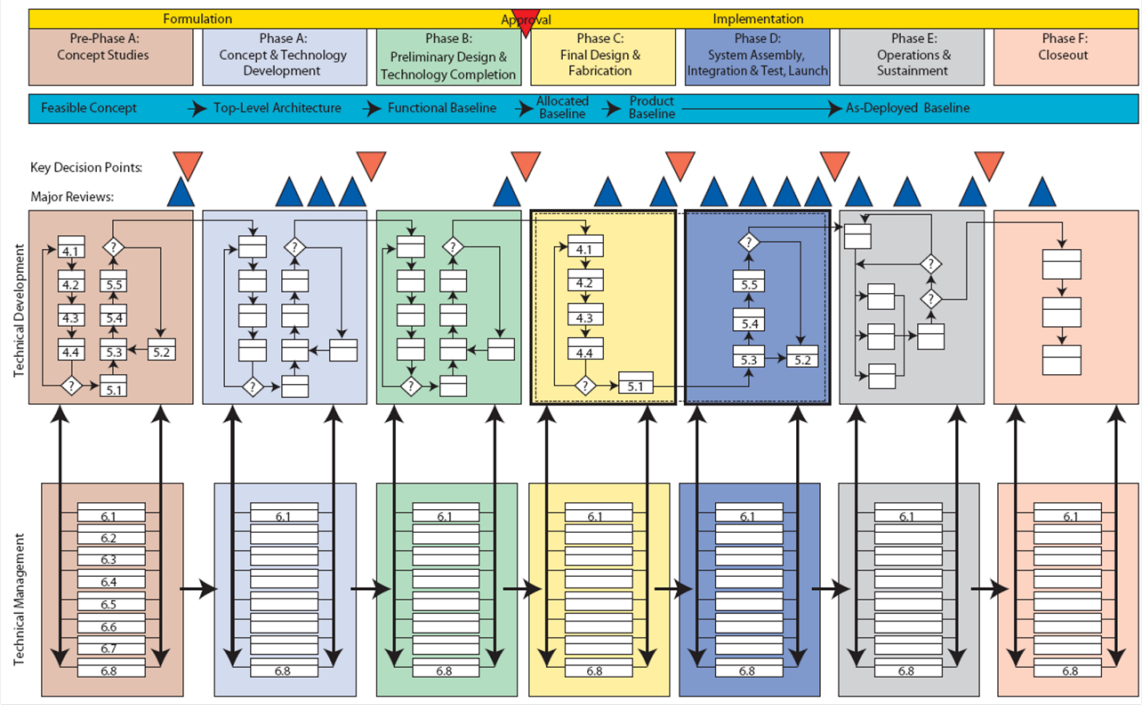


Figure 20: Simplified Version of NASA's Systems Engineering Project Life-cycle Process Flow Chart. Reproduced from [129]

Pre-Phase A: Concept Studies closely aligns with DoD's Pre-Milestone A, in that the general goal of this phase is to produce a many alternatives for potential programs from which to downselect, and assess all of these ideas for feasibility. The result of this phase is feasible system concepts, that can be carried forward to Phase A for further

refinement and study. At the end of Phase A, a clear system concept definition is produced. However, one major difference between the NASA approach and the DoD approach is that NASA assumes at the earliest phases of SE that a materiel solution will be used, and that this solution will be a new program [129]. Thus, the NASA SE Process does not include a stage for exploring the reuse of previous systems or other alternatives to achieving a goal. It may be that these types of solutions are less applicable to NASA and the harsh and challenging operating environment of space. Additionally, as the missions of NASA do not have an adversary, tactical or operational alternatives are not relevant. For these reasons, NASA assumption of a new program is very applicable to NASA, but makes their SE approach less applicable to the DoD. However, once a new program is launched within DoD, it may be worthwhile to explore NASA's SE Process further.

3.1.5 Systems Engineering Models for System of Systems

At first glance, it would appear that the dual vee model extends easily to the SoS level by simply adding another layer of depth at the top of the architecture vee to represent the system of systems, as shown in Figure 21. However, revisiting the key differences between a system and an SoS suggests that this may not be extensible as it appears, and it has been noted by many experts in the field that current systems engineering models are not appropriate for SoS [39, 181, 106]. While the larger scope, more complex integration, high degree of uncertainty and risk, geographic distribution of elements, and having elements which are not necessarily designed to fit the whole do not present any obvious challenges to Forsberg and Mooz's dual vee model, some of the other key differences present issues. In an SoS, elements may be evolving more continuously with elements of differing life cycles. This means that while any one evolution may be able to follow the vee process, there is no definitive start and end to the overall architecture development, and it would be impossible to line up

the entity vee for elements with differing life cycles. Dahmann et. al. states that the SoS characteristic of having operationally independent systems "challenges the traditional application of SE, since many of the models of SE are based on the ability of the systems engineer to define boundaries and requirements clearly and to control the development environment so that requirements can be optimally allocated to components based solely on technical trade analyses" [39]. Additionally, the dual vee model assumes that a single management or acquisition entity would be overseeing the overall architecture development. This is not necessarily true in an SoS. There are often many stakeholders, and further, there may be elements that are designed and built with a different purpose in mind and integrated into the SoS architecture after the fact. This means that the overall integration is not likely to follow a clean vee process. Additionally, with more ambiguous requirements and fuzzy boundaries, the verification and validation process for an SoS is very difficult. Finally, it is recognized that there is no concrete beginning and end in an SoS, thus the systems engineering is continuous and thus does not move cleanly through the vee.

For a single evolution of the SoS, however, it is likely that there will be only one (or a small number) of stakeholders, requirements which are more well defined with more well defined boundaries, and a definitive start and end to the acquisition and evolution process. Thus, for an evolution of a system of systems, or for the analysis to make a decision regarding how to evolve a system of systems, it is logical that a vee process would be appropriate.

Recognizing the need for a systems engineering model catered to the specific needs of systems of systems, several SoSE models have been proposed. In the Systems Engineering Guide for Systems of Systems, the DoD recommends an SoSE model that has since been nicknamed the trapeze model. The trapeze model was developed with a focus on acknowledged SoS [39]. The trapeze model is depicted in Figure 22. The trapeze model depicts what are identified by the DoD's Systems Engineering Guide

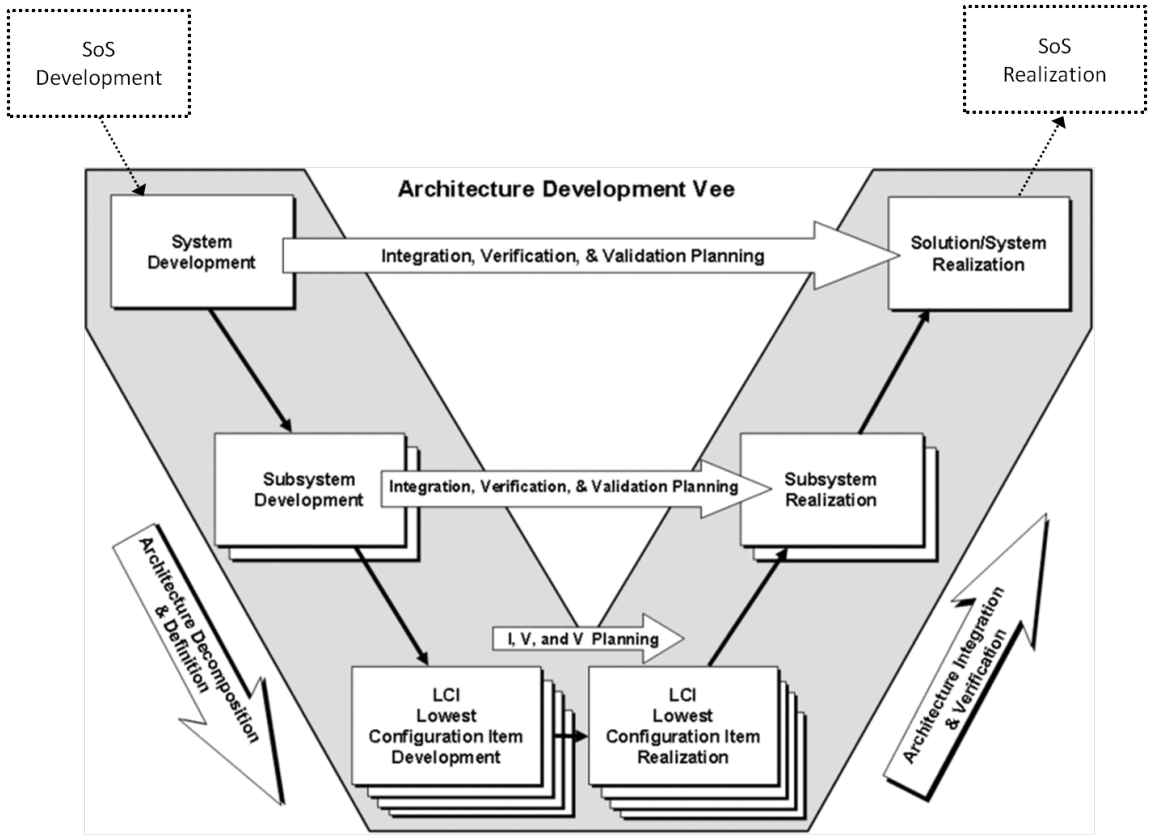


Figure 21: Forsburg and Mooz's Architecture Vee Model with Extension for SoS. Adapted from [66]

for Systems of Systems seven core SoSE elements and their relationships. The seven core elements are translating capability objectives, understanding systems and relationships, assessing performance to capability objectives, developing and evolving an SoS architecture. monitoring and assessing changes, addressing requirements and solution options, and orchestrating upgrades to SoS [57]. Since SE for SoS is recognized as being continuous, it is assumed that all of these elements are ongoing and thus there is no time frame placed on the model. However, for a single upgrade to the SoS, a vee model is used to manage that upgrade, and vees are also used to manage any system-level upgrades, as shown in Figure 23.

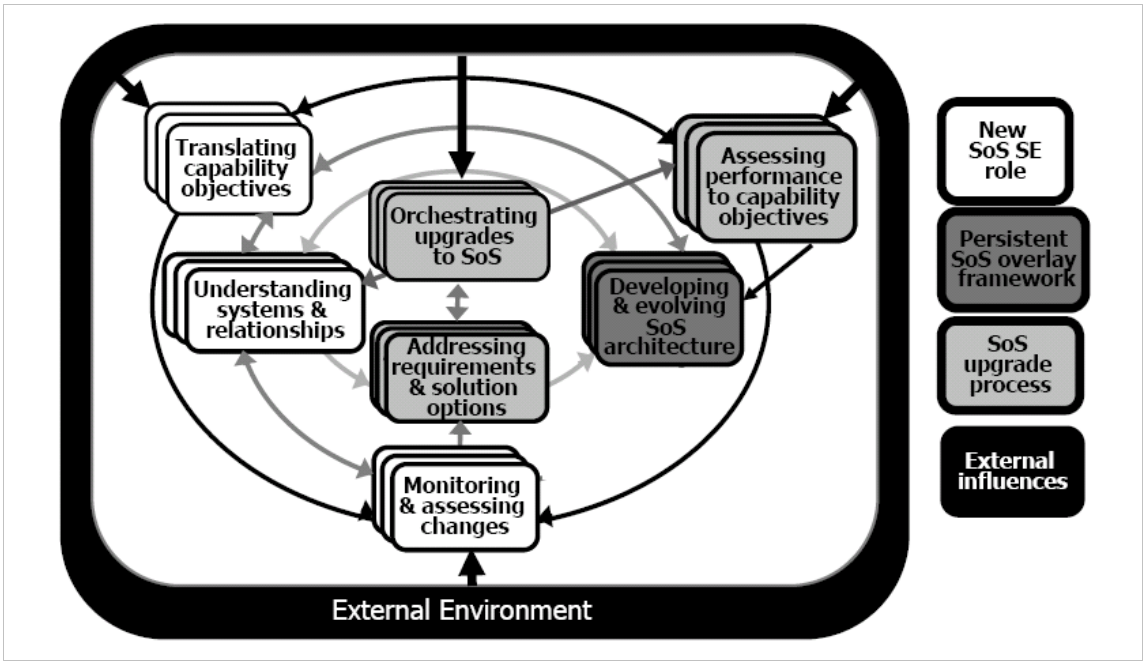


Figure 22: The Trapeze Model. Reproduced from [57]

3.1.5.1 The Wave Model for System of Systems Engineering

Dahmann et.al. have uncovered several shortcomings in the trapeze model, and have proposed the wave model as an alternate model for SoSE. While the wave model keeps many of the elements of the trapeze model, it is more geared towards practitioners. The evolution of the trapeze model to the wave model is shown in Figure 24. The wave model adds a temporal element to the trapeze model, recognizing that in the actual implementation and evolution of SoS, there are likely to be many phases of evolutions with associated time lines [39]. In DoD applications, the time lines of these evolutions are likely to be driven by the battle rhythm and operational needs [38]. It further attempts to simplify and clarify the relationships between the elements of SoS implementation and evolution, providing both a standard (or recommended) process as well as off-nominal processes that are likely to occur under certain circumstances. The wave model also has added a step to initiate the SoS. This does not necessarily mean that the SoS will be designed from scratch (although it would be possible), but

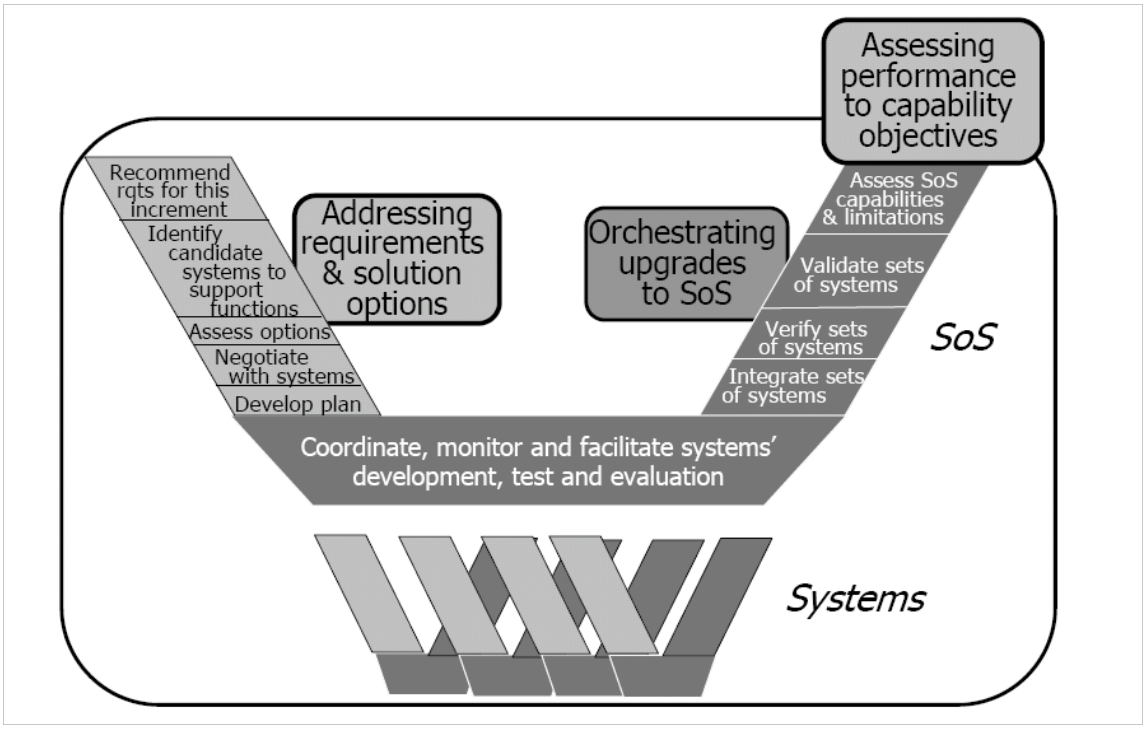


Figure 23: DoD Systems Engineering Guide for System of Systems Engineering Vee Model for a Single Increment [57]

rather refers to the point at which the SoS becomes acknowledged as an SoS and begins to be managed as such. The initiation of the SoS includes understanding and getting stakeholders to agree upon the high-level capability objectives for the SoS, identifying the roles and expectations of users and stakeholders, an understanding of the CONOPS for the SoS, and gathering information about any systems that are likely to affect the capability objectives [39].

Once the SoS has been initiated, initial analysis is conducted. This considers the current state of the 'as is' architecture for the SoS and provides the baseline for SoS development and improvement. This assumes that the capability is currently realized in some way, but does not necessarily imply that the current realization is a sufficient implementation, or that it conforms to principles for effective SoS. At this stage, the technical baseline is developed, performance measures are established, performance data are gathered, the requirements space is established, and SoS planning elements

are identified, including risks and mitigations [39].

This initial analysis should provide the information required to create a plan to develop or evolve the architecture. The primary output of this phase is the proposed architecture, but also includes the migration and evolution plan necessary to realize that architecture. Key risks are identified and a mitigation plan is developed. The SoS architecture developed here should include the systems, functions, relationships and dependencies, end-to-end functional flows, data flows and communication protocols. This step will develop the requirements space for the implementation, including changes in systems, interfaces, and functionality. If a new system acquisition is required, the requirements for the new system would be developed in this stage [39].

Once this is done, the SoS update is planned. This evaluates the priorities of the proposed changes, and determines which updates to include in the current update cycle and which, if any, need to be postponed to a future update cycle. The allocated baseline for the update is created and risks and risk mitigations are identified. In addition, integration and testing plans are created, and key stakeholders must come together to reach an agreement on the plan and implementation schedule. It is important that the schedule created for the SoS take into account the development and maintenance schedules of constituent systems [39].

Finally, the actual implementation of the update takes place. This includes the implantation of the proposed changes and testing at the system and SoS level. The output of this phase is a new baseline for the SoS, which will feed into the next wave of the model. The next wave begins by continuing the SoS using the new baseline, and revisiting the current state and plans for evolution of the SoS. Once again, ideas for potential updates to the SoS can include anything from changes to the CONOPS, system-level changes, interface updates, or addressing unexpected factors resulting from the previous update. This leads to a new evolution of the architecture, which is

then planed and implemented, feeding into the next wave, and the process continues [39].

This model has the advantage of being designed to handle an SoS and the associated challenges. It is an ongoing model, which addresses the concerns of continual evolutions and continuous systems engineering. The wave model is more able to deal with operationally independent systems with differing life cycles than standard system engineering models, because updates can occur over several waves, and these can be timed to coincide with existing maintenance schedules for the various systems. The model has several places where stakeholders come together to reach an agreement on both the expectations for the SoS and the evolution plan, which addresses concerns about the impact of multiple stakeholders. Another advantage of this model is that the wave approach allows updates to be implemented over a series of evolutions, which means that model verification and validation can be done in between each wave. After each wave, assessments can be made as to whether the models adequately predicted the performance, cost, risk, and schedule for the previous wave, and models can then be updated to include lessons learned. In this way, the modeling will continue to improve over time, and confidence in the overall evolution plan can increase with each successive wave. It is also interesting to note that each wave resembles the entity vee model, reinforcing the idea that while the vee model is not appropriate for SoSE, it can be appropriate for a single update increment. It makes sense that a vee would be appropriate, since each wave has a clear starting and ending point with established requirements. A mapping between a single wave and the entity vee is shown in Figure 25

It is also interesting to consider how the wave model for SoSE fits in with the overall DoD acquisition process. Given that the acquisition process was really designed with the acquisition of a single system in mind, there is no natural alignment with the acquisition milestones. However, examining the information requirements at

each step throughout the acquisition process can provide some guidance as to how the wave model would be able to work with the current acquisition system. The SoS level analysis to determine capability needs and explore architecture-level changes within the DoD occurs primarily during the CBA process. It is during this process that non-materiel solutions are explored, and decisions are made as to whether or not a new acquisition is required. Therefore, it would make sense that the steps 'Conduct SoS Analysis', 'Develop/Evolve SoS Architecture', and 'Plan SoS Update' would occur during the CBA process.

However, there are some key distinctions between the analysis required for a CBA and the analysis suggested by the wave model. CBAs are typically done for a single mission area, while the wave model recognizes that the SoS is likely to be multi-mission, and recommends that analysis is done across multiple mission threads. Secondly, CBAs are typically a one-time process to justify a particular system acquisition. The wave model proposes that the analysis done in the CBA should be ongoing and continually updated as evolutions to the SoS occur. The wave model also accounts for the possibility of having multiple solutions that evolve over time. For example, if it is determined that a new aircraft is required, the acquisition of this aircraft can take as long as 20 years. However, as the capability needs are persistent during these 20 years, a stop-gap solution leveraging existing systems is needed in the interim. The wave model allows for an evolution plan that implements the stop-gap solution while starting the acquisition of the new system, and is able to assess the progress of the acquisition through multiple waves and continually determine if this new system is still required, or if changes may have occurred that either change the requirements of the acquisition or negate the need for this new system. In this way, the wave model would allow for decisions made during CBA to be revisited as the operational landscape changes. If the wave model is accepted as a backbone for the overall SoSE

process, a new requirement is generated for a CBA methodology. The CBA methodology should include the development of a dynamic CBA environment that can be easily updated with each wave and enable continual analysis of the SoS over time. This implies that the products of the CBA should not be a paper document, but instead should be a framework to support current and future decision making that preserves the results of previous analyses, can be easily updated, and allows the CBA process to be repeated very quickly once updates are implemented and a new baseline is developed.

While the model provides a solid backbone for SoSE and addresses the specific challenges of an SoS, it still lacks guidance on how to perform the analysis, how to determine the development and evolution of the architecture, and how to decide which potential evolutions are appropriate in the context of time lines and budgetary constraints for each wave. The methodology developed as part of this research will attempt to address how this analysis and decision making can be done in support of the wave model and the existing CBA process. Furthermore, the method will aim to result in the creation of a dynamic framework that supports continuing analysis across multiple waves.

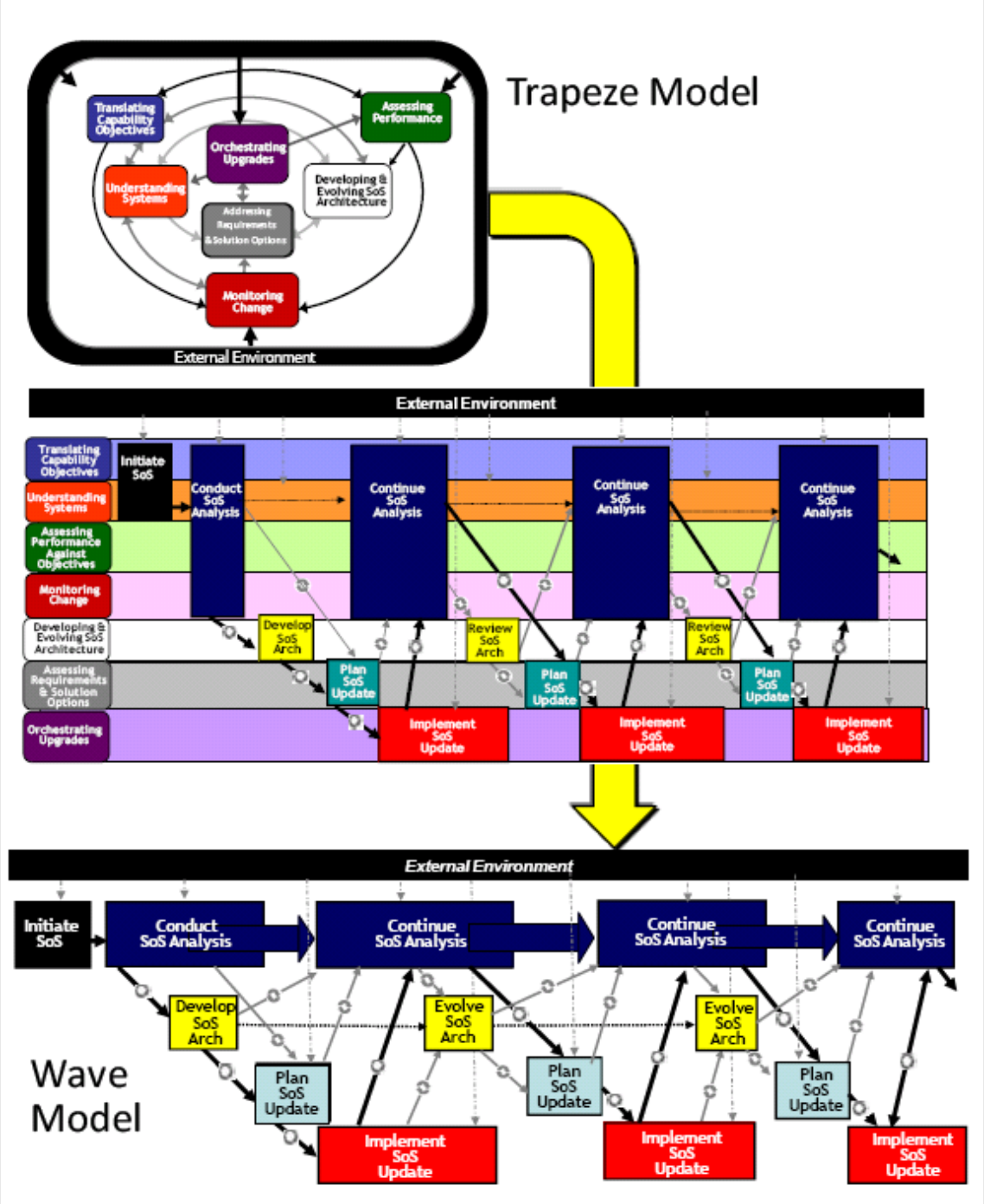


Figure 24: The Trapeze Model being 'Unwound' to create the Wave Model. Reproduced from [39]

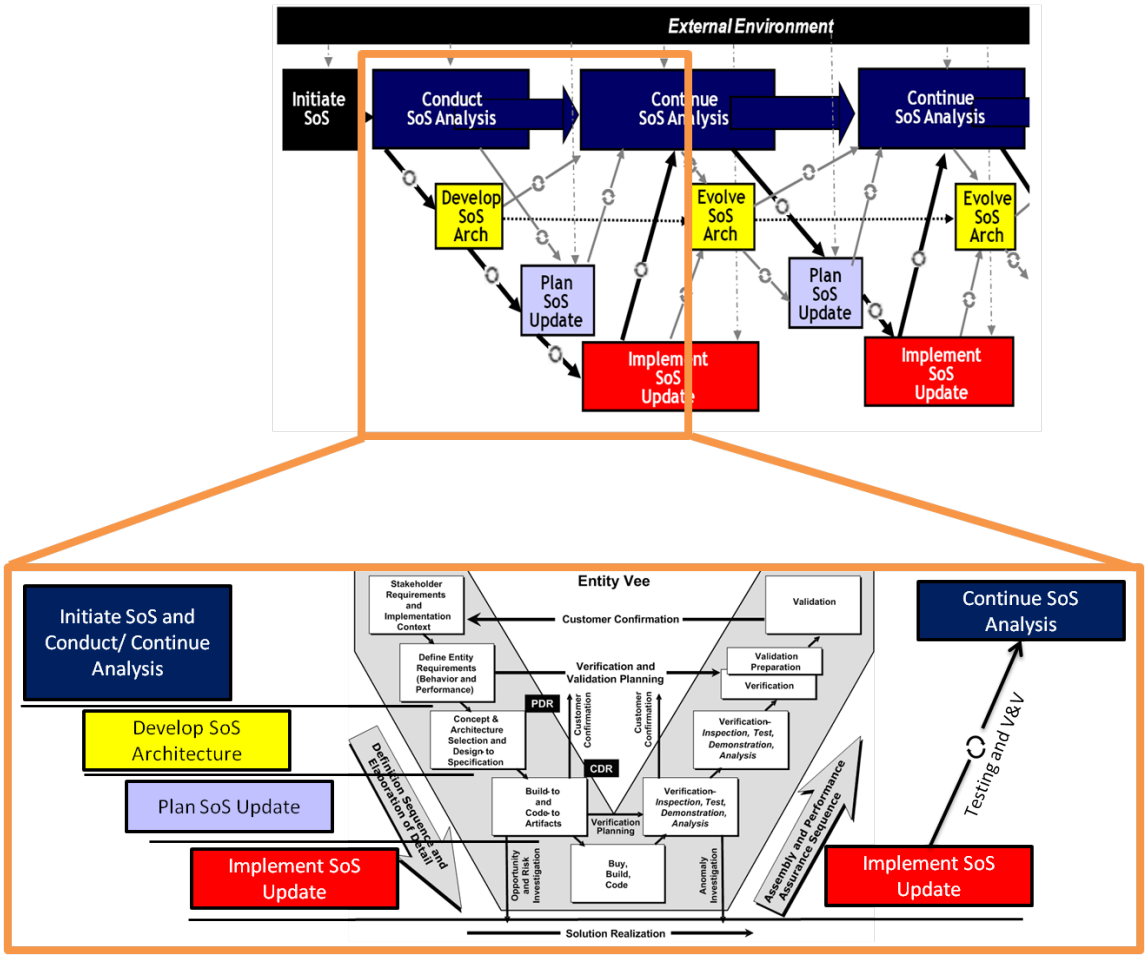


Figure 25: Mapping the Wave Model to the Entity Vee. Figure based on [39], [66]

3.2 Existing Techniques for Metrics Derivation

There are several system engineering approaches for deriving the measures or metrics necessary to assess the feasibility and viability of proposed solutions to a problem. In version 1.0 of the INCOSE Systems Engineering Measurement Primer, two approaches are suggested for metrics derivation, the Practical Systems/Software Measurement (PSM) approach and the Goal Question Metric (G/Q/M). These are both summarized here. However, in version 2.0 of the Primer, the G/Q/M approach was removed from the guidance, although no explanation is offered as to the reasons behind its removal. The current guidance for determining measures consists of seven steps. The first step is to align measurement with organizational needs. This refers to determining the information needs at high levels in the organization. The next step is to identify and prioritize information needs. This step includes gathering the project management team and stakeholders to identify and prioritize information needs specific to the project. Eight categories for information needs are suggested, including schedule and progress, resources and cost, system performance, growth and stability, product quality, life cycle process, technology effectiveness, and customer/user satisfaction. Specific information needs in these area are then identified based on different sources of input, which include project risk analysis, project constraints and assumptions, product acceptance criteria, known project problems, project goals and objectives, and external requirements or dependencies. A set of criteria weights are then formulated to reflect the relative importance of each criterion to decision makers. Next, measures are specified that satisfy information needs. PSM is recommended for this step [91].

3.2.1 Practical Systems/Software Measurement (PSM)

The Practical Systems/Software Measurement (PSM) method relies on the identification of a common set of project or process issues, such as schedule and progress,

resources and cost, and system performance, among others. These common goals are used as a basis for the identification of more specific project issues, which are then mapped to candidate measurement categories and appropriate candidate measures. Candidate measures are selected from a variety of standard sources, such as the INCOSE Handbook or other standards. The candidate measures are then evaluated against a set of selection criteria and this is used to select the measures to be used. The Primer emphasizes the importance of specifying measures in an unambiguous manner. The next several steps of the process involve data collection, analysis, storage and reporting; the criteria for evaluating the measurement process; the allocation of resources to measurement; and finally, the acquisition of supporting tools and technologies. As these steps are outside the scope of applicability to this element of this research, they will not be discussed here [90].

The INCOSE Handbook gives the following criteria for a good metric [89]:

- It tells how well organizational goals and objectives are being met through processes and tasks.
- It is simple, understandable, logical and repeatable.
- It shows a trend, more than a snapshot or a one-time status point.
- It is unambiguously defined.
- Its data is economical to collect.
- The collection, analysis, and reporting of the information is timely, permitting rapid response to problems.
- The metric provides product and/or process insight and drives the appropriate action(s).

The SE measurement primer, also from INCOSE, has a similar list. It says that good measures are relevant, complete, timely, simple, cost effective, repeatable, and

accurate [90]. If these lists are combined, a good measure would be relevant, complete, timely, unambiguous, logical, simple, cost effective, repeatable, and accurate. This list will provide a basis for assessing the utility of measures chosen for study.

3.2.2 Goal Question Metric (G/Q/M)

Another systems engineering approach for metric and measure derivation is the Goal Question Metric (G/Q/M) approach. The G/Q/M approach consists of four basic steps. First, an information goal is identified. Next, a set of questions are developed to evaluate if the information goal is being met. Third, the measures, both direct and indirect, required to answer the questions are identified as well as the means to collect them. Finally, the measures are applied and their usefulness is evaluated. If they were unable to fulfill the information goal fully, new measures are selected and the process is repeated [90].

3.3 Existing Techniques for Gap Analysis

Gap analysis is, in general, the process of clarifying the difference between the current or existing state of something and the desired state of something. For CBA, this includes both the identification of gaps in capabilities and overall mission performance, and the ranking of those gaps. The ranking of gaps can be a function of the size of the gap, the operational impact of the gap (the criticality), or both. A third dimension that is often neglected is the difficulty, cost, and risk of filling the gap. The reason that this dimension is often ignored is that the Analysis of Alternatives (AoA) is performed downstream of the gap analysis, and the information is never brought back to the gap analysis. At the end of the CBA process, an approach is chosen based on its ability to fill selected large, highly critical capability gaps. It should be recognized however, that a solution may be to accept the operational risk associated with not filling a gap if the difficulty, cost, and risk outweigh the benefits. [101]

An additional challenge is that the size of a gap can be characterized in several

ways. The gap size can be the difference between the current performance and some minimum threshold performance, the difference between the current performance and the ideal desired performance, or the difference between the current performance and the performance attainable with the best-performing alternative. The first measure of size (the difference between the current performance and some minimum performance threshold) can be effective, particularly in cases where improving performance beyond the threshold value does not significantly impact the ability to perform the mission. For example, if the mission were to drive to work as quickly as possible, both a person with a small compact car and a person with a sports car could perform this mission equally well. The person with the sports car would not gain an added advantage based on having a higher top speed as compared to the small compact car, because the actual minimum drive time would be limited by the speed limits on the road. In this case, a person with a bicycle may have a capability gap, because the bicycle is not able to attain the performance threshold of the speed limit. However, in cases when improvement beyond the threshold would provide considerable performance to mission capability, this extra improvement in performance is not captured, although it may be highly desirable.

The second measure of gap size (the difference between the current performance and the ideal desired performance) is useful in analyses aimed at long-term future performance. In these cases, some ideal goal has been set for the future, and the current performance is compared with that goal. Alternatives are measured by how well they are able to close the gap between current and desired performance. This performance can either be treated as a threshold or as a goal. However, the difficulty in this method is determining what level of improvement is acceptable if no solutions are able to meet the goal. It is possible to combine these two ways of formulating the problem when performing gap analysis. Capabilities where a performance threshold is required, but additional improvement will not increase mission performance can

be assessed based on the difference between the current capability and the threshold, and capabilities in which improvement beyond a threshold will continue to improve mission performance can be measured against an ideal performance goal. The third measure of size (the difference between the current performance and the performance attainable with the best performing alternative) is very difficult to measure, because the alternative space would have to be developed and analyzed for all potential capability gaps prior to selecting which capability gap to pursue closing. Since it is likely that not all alternatives are currently existing solutions for which there are existing performance data, it is likely that a significant amount of analysis will be required to support this effort. When there are a large number of diverse gaps to be assessed, the amount of analysis required to support this technique becomes unfathomable. Thus, this technique is most appropriate when the number of capabilities is small and the number of alternatives is also small.

When performing gap analysis, it can be difficult to determine how to assess current and desired performance. The performance of different capabilities is likely to be measured using different metrics. It may therefore be necessary to use a normalized scale to compare gap size and criticality. Additionally, the criticality of a gap may be extremely difficult to assess quantitatively. The actual measurement of a gap size or a gap criticality can be done in several ways. Literature search combined with subject matter expertise can provide information about the current state-of-the-art as well as the desired level of a capability. In the case where subject matter experts (SMEs) are used, it is a common practice to use a qualitative scale to obtain feedback. Qualitative answers are then mapped to a simple numerical scale for analysis. An alternate method, in the absence of existing data or subject matter expertise, is to create a modeling and simulation environment to obtain the required information. In this case, it is necessary to determine the set of performance parameters that will be assessed in the modeling environment to determine gap size.

It is also important to realize that gaps can exist in areas other than performance. For example, the current solution to a capability need may adequately perform the capability, but the cost or risk associated with the solution is unacceptable. This represents a different kind of capability gap that is important and cannot be overlooked in gap analysis.

Despite the challenging nature of performing gap analysis, gap analysis during CBA is often done in an ad-hoc manner. [172] When more rigorous gap analyses are performed, the methods and tools used are not published in the open literature to enable future gap analyses. In the Chief Information Officer Council's *A Practical Guide to Federal Enterprise Architecture*, a detailed gap analysis is called for, but the guidance on performing gap analysis is only two short paragraphs. Furthermore, the gap analysis recommended in this guide only considers the difference between two architectures, the as-is architecture and the to-be (or target) architecture, across all architecture products. This assumes that the to-be architecture is known, which is not a valid assumption for CBA applications. [35]

Gap analysis is commonly performed in the area of ecology and conservation. These gap analyses are performed to determine where conservation efforts fall short in providing the right habitats, vegetation, and terrain to protect endangered species and to reduce the risk of new species becoming endangered. The gap analysis methods used in this community are well documented, but the steps of the process and recommended tables and figures are very specific to the community. However, the steps of the method are generalizable. First, needs are identified in each category. These are based on the threshold values that are required for species preservation and survival. Next, the existing coverage in each category is compared with the needs, showing where needs are not met. The criticality of needs are assessed based on the endangerment level of the species. Then, statistical analysis is used to identify the probability of closing that gap with the current management plan. This helps identify

where gaps are currently being addressed but have not yet been closed vs. those that have not received attention. Next, any additional needed analyses are performed, based on the specific needs of the study. Finally, the results are incorporated into a report. One point that is emphasized is the importance of focusing on the needs of all species together, and not treating the needs of one species at a time. This prevents a reactive mentality where gaps are only closed once species become endangered (which requires more expensive management plans), and helps to prevent species that are not endangered from becoming endangered. [156]

While the application of this gap analysis method is very different from CBA, the general process and lessons learned can still be applied. Critical elements to the gap analysis will include a comparison of desired performance with current performance, an assessment of whether current in-work programs will be able to close the gap, an assessment of the criticality of the gap, and statistical analysis to account for uncertainties in the analysis. In the same way that a multi-species approach is required in conservation, a multi-mission, multi-service approach is required in defense applications. This will allow decision makers to better understand how potential solutions can close gaps across the enterprise, and help to avoid a reactive mentality where only current gaps are considered and potential future gaps are ignored. This would help avoid situations where gaps in any one mission are ignored because that mission is not critical in current engagements, even if there is potential for it to become very critical in potential future engagements. However, since the specific tables and frameworks used to execute the general method are specific to the ecology applications, it is necessary to find an alternative framework with which to conduct the gap analysis. For this research, the Relational-Oriented Systems Engineering Technology Tradeoff Analysis ROSETTA framework will be used. The application of ROSETTA to gap analysis is discussed in more detail in Section 4.2.2.

3.3.0.1 Quality Function Deployment

One form of qualitative, SME-driven analysis is the quality function deployment (QFD). The QFD is made from combining the seven Management and Planning Tools and is often used in the early phases of Systems Engineering to elicit tacit knowledge from SMEs and use that knowledge to identify key design drivers and trades. QFD is a proven quality engineering technique that has been credited with significant reductions in cost and product development time, as well as increases in productivity when applied correctly in product development [147]. QFD leverages information elicited from SMEs to create a mapping between customer needs and requirements and the engineering characteristics that are required to fulfill those needs. The goal is to understand the relationships between the engineering characteristics and between the requirements, and to identify which characteristics are most important to fulfilling the top level requirements. QFD provides a methodical and clear translation between the less technical 'voice of the customer' and the more technical parameters of the engineer. QFD has been credited with the quality revolution in microelectronics and automotive sectors [25, 81]. Because of this success, the application of the QFD has since been repeated in many other product development sectors, including consumer electronics, home appliances, clothing, and many others [81]. The QFD has also been applied to a variety of other topics beyond product development, including the link between manufacturing flexibility and market requirements [134] and even university course design [119]. However, because QFD originated in manufacturing, where relationships have been observed to be linear, the relations in the QFD are also linear. Case studies in the literature have not verified that this assumption applies before using QFD, thus calling into question the validity of these applications of QFD and the associated results. For SoS, which are often characterized by non-linearity, this assumption will not hold, and thus an alternative approach to QFD is required.

A QFD is performed by consulting with SMEs and customers to create one or

more HOQ. The HOQ is the basic tool enabling the QFD approach. The execution of the QFD includes a set of planning and communication routines that help develop and use these HOQs, which are called the Seven Management and Planning Tools. The HOQ is the conceptual map that provides the means for planning and communication between customers and engineers, two groups of people with different problems and responsibilities [81, 25]. The key parts to a HOQ are shown in Figure 26. The requirements are given in part 'A', and the engineering characteristics in part 'E'. The creation of the HOQ begins with elicitation of the customer requirements and the engineering characteristics which will measure the fulfillment of the requirements. The customer then puts weightings on the requirements (often on a linear 1-10 scale) in part 'B'. Next, for each engineering characteristic, the direction of improvement ('larger is better', 'smaller is better', 'nominal is better') is identified in part 'D'. The body (part 'C') of the HOQ, is populated by asking the SMEs to create a mapping between the requirements and the engineering characteristics. This mapping describes how strongly each engineering characteristic is related to each requirement. In other words, the SMEs are answering the question, 'In general, how much impact does this engineering characteristic have on the ability to meet this requirement?' Typically, this question is answered with the following: ''strong impact', 'medium impact', 'weak impact', or 'no impact'. The qualitative answers are then translated to a non-linear numerical scale. One possible scale has 9 corresponding to 'strong impact', 3 corresponding to ''medium impact', 1 corresponding to 'weak impact', and 0 corresponding to 'no impact'. The choice of scale is dependent on the user and the application of the HOQ. It is common, however, for the chosen scale to be non-linear, such that a strong relationship carries significantly more weight than a weak one. For each engineering characteristic (each column of the HOQ), its total importance is calculated by multiplying the weighting of each requirement by the impact score for that engineering characteristic to that requirement, and then summing the product.

This is done for each engineering characteristic, and the results are given in section 'I' [25].

Since it is possible, or even probable, that the engineering characteristics and requirements may be correlated to one another, these correlations are captured in the HOQ using triangular mapping matrices on the side and top of the HOQ. The correlations between requirements are given in part 'H', which is sometimes referred to as the 'greenhouse'. The correlations between engineering characteristics are given in part 'G', which is sometimes referred to as the 'roof'. These correlations are elicited from SMEs, and each identified correlation is evaluated as 'strong negative' (often corresponding to a score of -3), 'negative' (often corresponding to a score of -1), 'positive' (often corresponding to a score of 3), or 'strong positive' (often corresponding to a score of 9). These identified correlations can be used to help identify where critical design tradeoffs may be expected to occur. Although numerical scores are associated with the correlations, these scores are not typically used in the calculation of the QFD results. In order to allow for benchmarking, alternative concepts are enumerated in part 'F' of the HOQ, and evaluated with respect to how well they meet each of the requirements [25].

The implementation of QFD often requires the hierarchical linking of several HOQs. This linking is performed by making the engineering characteristics from one HOQ the requirements for the next HOQ. The requirement weightings for these engineering characteristic requirements are then the output scores on the engineering from the first HOQ. The greenhouse for the second QFD is the same as the roof of the first QFD. The engineering characteristics used for the second QFD are at the next lower level of detail from the original engineering characteristics. SMEs are then again consulted to develop the mappings and aid in the associated voting, although it is likely that a different set of SMEs will be required. This deployment process can be repeated until the desired level of fidelity is achieved [25]. The general QFD

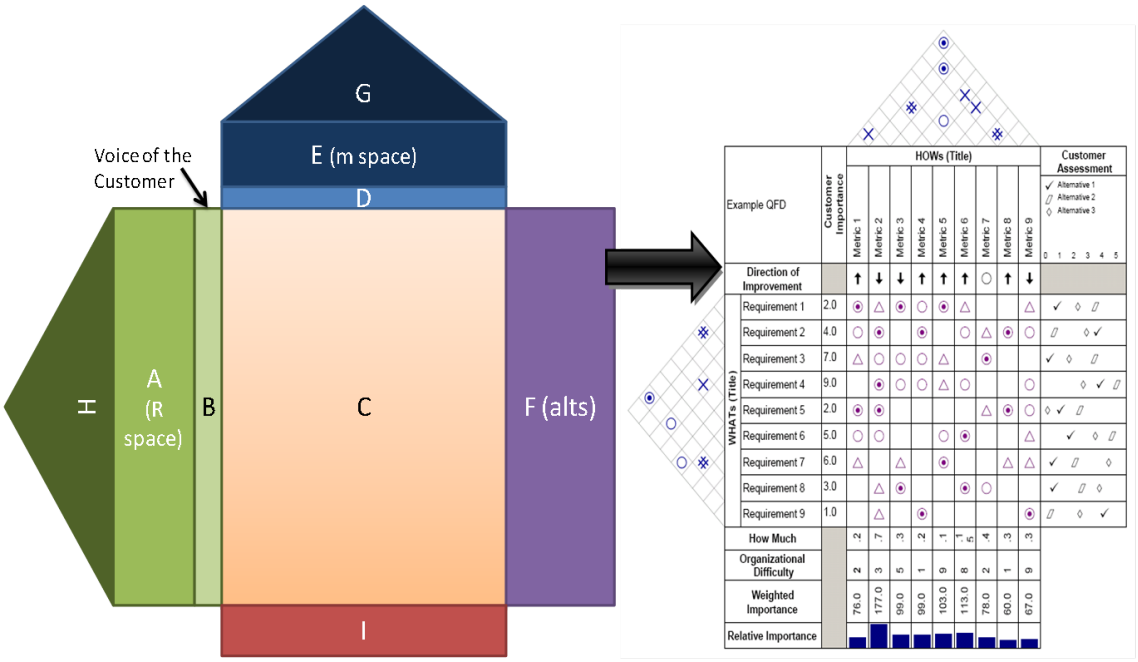


Figure 26: Generic HOQ Structure and Example HOQ. Reproduced from [118]

process discussed above is summarized graphically in Figure 27.

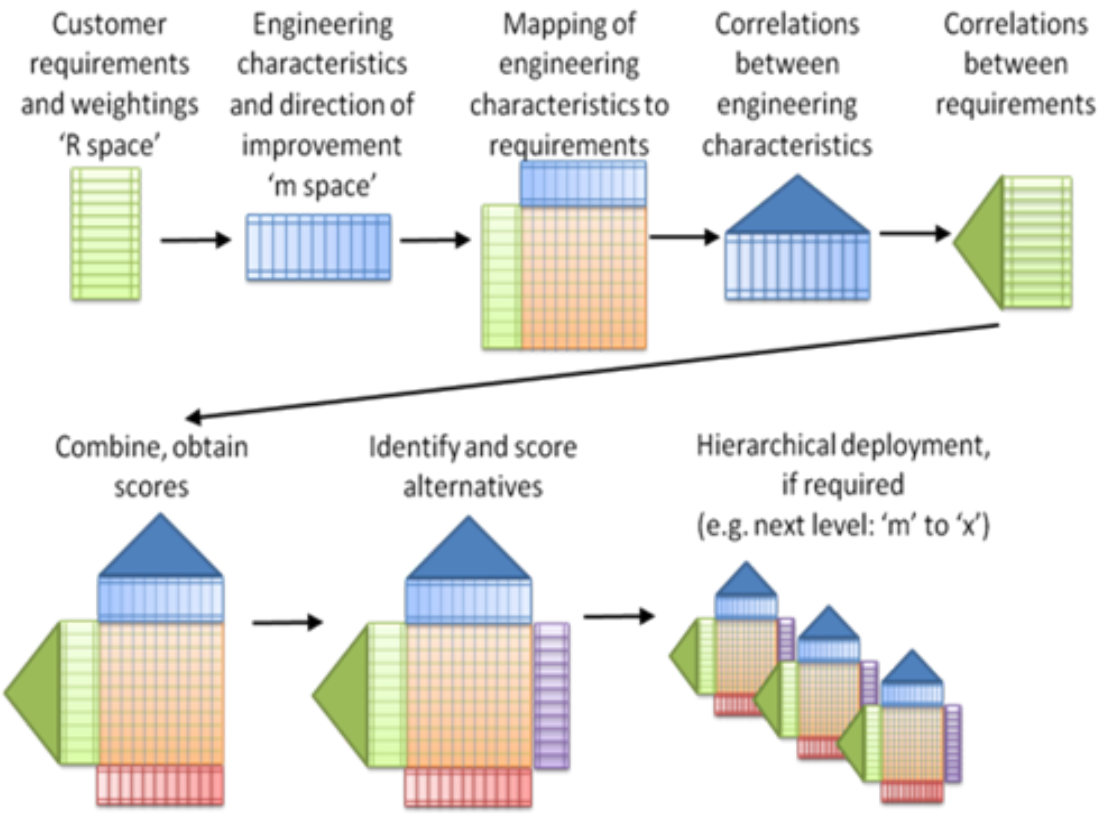


Figure 27: General Process for QFD. Reproduced from [118]

3.3.1 ROSETTA

The Rosetta stone provided a means to translate between the Greek, Hieroglyphics, and Egyptian demotic languages by having the exact same text (a decree) repeated in all three languages. In the same way, the Relational-Oriented Systems Engineering Technology Tradeoff Analysis (ROSETTA) framework has been developed to provide a means for translation between theoretical mathematics, subject-matter expert driven analysis, and modeling and simulation by representing a single problem using all three types of analysis and highlighting the commonalities and differences resulting from the different representations of the problem. ROSETTA also provides a structured means for fusing qualitative and quantitative data in engineering analyses. ROSETTA was initially introduced in [118], and has been expanded here.

ROSETTA was inspired from the exploration of common techniques for qualitative and quantitative analysis in systems engineering and technology tradeoffs. The QFD was explored as a commonly used tool for qualitative, SME-driven analysis in systems engineering and early-phase design. The information obtained in the QFD qualitatively can able be obtained quantitatively and more objectively by using modeling and simulation. This modeling and simulation is often performed in later phases of the design process. In the ROSETTA framework, the behavior of the modeling and simulation environment in the area of interest will be captured by using the response surface method (RSM) [122] or an alternative surrogate modeling approach to wrap a set of surrogate models, around the modeling and simulation environment. Because the surrogates are simply a functional regression of the model, these simple functions can be used to explore the behavior of the modeling and simulation environment quickly and thoroughly. These surrogates will enable virtually instantaneous execution of the modeling and simulation environment, thus allowing trends across the design space to be easily understood and quantified. This aids in the determination the strengths of the relationships represented in the body of the HOQ. The surrogate approach also enables a rapid execution of a Monte Carlo Simulation (MCS), and the results of this can then be used to obtain information about the response space, such as the correlations found in the roof and the greenhouse of the HOQ.

Having noted the similarities between the modeling and simulation and SME-driven analyses, a more formal understanding of the relationships and the resulting analysis can be obtained by examining the mathematical nature of the relationships. A mathematically-based viewpoint of HOQ that applies rigorous transformation formulae to describe the relationships in the HOQ is developed to understand the nature of the HOQ and its implications in the overall engineering approach. This mathematical viewpoint can also help determine what information is shared in qualitative

and quantitative approaches, and how this information can be used in a complimentary fashion. This mathematical approach can provide a direct translation between the information contained within the modeling and simulation and the information elicited from SMEs.

In order to more easily explain the application of ROSETTA to a technology tradeoff, the following terminology will be used: Customer requirements will be considered the future goals that a chosen technology portfolio is trying to meet, and will be referred to as the 'R-space'; the engineering characteristics are the metrics on which the technologies will act, and are called the 'm-space'; and the fundamental design variables of the system that will map to the metrics at the next lowest level will be called the 'x-space'.

There are four general statements that are critical to ROSETTA:

1. Existing M&S methods and QFD already use transformations
2. QFD provides a qualitative approximation of the same information that is obtained quantitatively in M&S
3. ROSETTA applies ROSE to QFD and M&S, and allows these transformations to be formalized
4. The formalization of these transformations allows improvements in how these existing methods are used to support engineering decision making

3.3.2 Translation Between Qualitative and Quantitative Analyses

The QFD approach described in the preceding sections leverages a subjective approach for estimating the relations between customer requirements, metrics, and design variables. These same relationships are commonly developed through modeling and simulation (M&S) in subsequent design phases. In order to do this, the requirements must be measurable and a function of the metrics. If this is not the case, it may

be necessary to move to a lower level in the deployment. The metrics are a function of the independent design variables, which would be captured qualitatively through QFD deployment to a second level. This suggests that when performing QFD in initial stages of the engineering process, it is useful to decompose the QFD in terms of measurable quantities that are a function of the previous level of the decomposition. That is to say, the requirements should be functions of the metrics, and the metrics should be functions of the design variables. All of the metrics that influence all of the requirements must be included in the HOQ in order to fully capture the relationships and rank the metrics fairly. The second HOQ deployed in the QFD maps metrics to independent design variables, and again, must include all of the design variables that impact the metrics. The approach presented here to map the QFD and M&S was published by the author in [118].

In order to obtain both the relationships between the requirements and the metrics vis M&S, as well as between the metrics and the design variables, two parallel M&S setups are developed. One is dedicated to each set of relationships contained within each HOQ. Since producing the needed information requires a very thorough exploration of the full design space, a parametric environment is needed to fully capture the behavior within ranges of interest. Since development of a parametric environment will require large numbers of cases to be run, surrogate models will be made from the actual codes to speed up the analysis process. The surrogates are extremely accurate to the behavior of the M&S environment and run virtually instantaneously. More importantly, the surrogates provide a continuous function representing the design space, which can be used to visually and mathematically explore and understand the design space. If possible, Response Surface Equations (RSEs) are the preferred form of these surrogates. They are simple to create and relatively intuitive, giving a more intuitive mapping of the relationships. However, any form of surrogate will work for this approach. The general form of a second-order RSE with k independent

variables is shown in Equation 1 [122] where the error term ε is assumed to have a normal (N(0,1)) distribution.

$$R = b_0 + \sum_{i=1}^{k} b_i x_i + \sum_{i=1}^{k} b_{ii} x_i^2 + \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} b_{ij} x_i x_j + \varepsilon \quad (1)$$

To create surrogates an appropriate design of experiments (DoE) is selcted and executed using the M&S codes. Then, the results of this subset of cases is used to perform a regression, which creates the surrogate models. The form of the DoE is chosen based on the expected form of the surrogate models Guidance to choosing a DoE is well documented in other sources, and thus will not be discussed here. If the assumed type of surrogate model does not meet the desired accuracy, it may be necessary to try an alternate form of model, which may require additional DoE runs. The surrogates are used to describe the relationships between independent variables and the responses. These relationships can be visualized using a prediction profiler, such as the one shown in Figure 28. The prediction profiler which captures all of the sensitivities between the metrics and the requirements, and can be used to estimate the partial derivatives. The surrogates can also be used to run a Monte Carlo Simulation (MCS) by placing random uniform distributions on each of the independent variables and running a large number of cases (often 10,000 or greater). The MCS covers the full design space and can be used to obtain the correlations between the responses using multivariate analysis. This is done using a multivariate plot, similar to the one shown in Figure 30.

Obtaining the sensitivities and correlations between the metrics (contained in the roof of the HOQ) requires that this process be repeated at the next level of detail, treating the metrics as the responses and using a new set of independent variables that will influence the metrics. This is equivalent to the second deployment of the QFD. Repeating the process for this new set of variables yields the correlations and sensitivities between the metrics. The full M&S approach used in this research is

summarized in Figure 29.

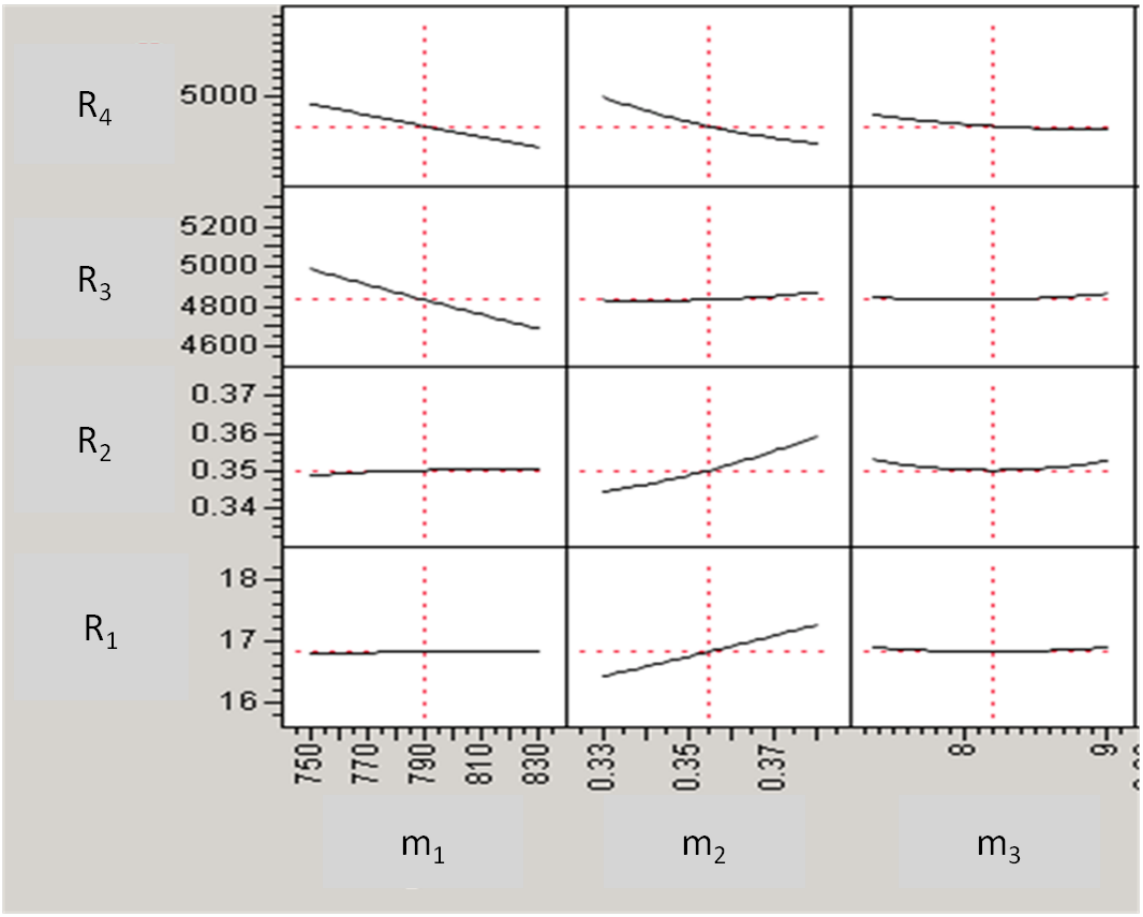


Figure 28: Example Sub-section of a Prediction Profiler. Adapted from [118]

Understanding this quantitative approach to obtaining the relationships captured in the QFD highlights several important points. First, visual examination of the multivariate plot (demonstrated in Figure 30), suggests that correlations alone are an incomplete description of the relationships between requirements or between metrics. Correlation gives a measure of the linear dependence between two variables, and is often expressed on a -1 to 1 scale using Pearson's correlation coefficient. The equation for the correlation coefficient is given in Equation 2.

$$\rho_{X,Y} = \frac{Cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \tag{2}$$

A correlation coefficient close to -1 implies a very strong negative correlation. A

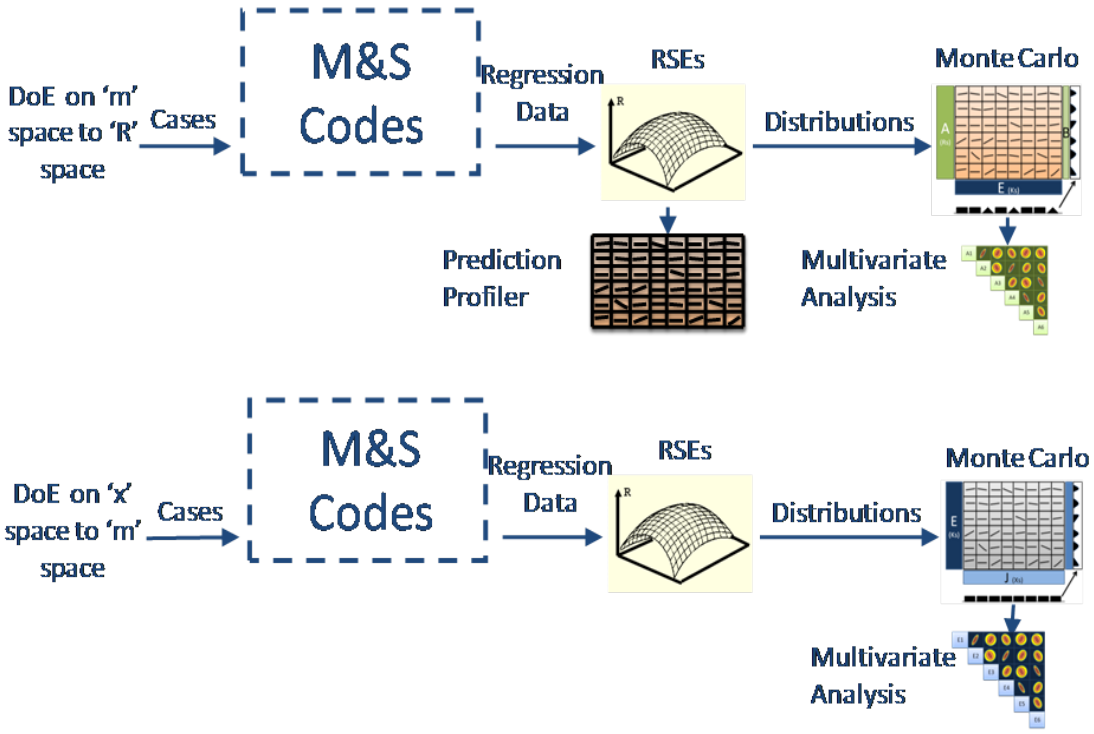


Figure 29: ROSETTA Modeling and Simulation Analysis Setup. Reproduced from [118]

value close to 1 implies a strong positive correlation. A value close to 0 implies no correlation. However, this only measures the linearly of the relationship, and not the strength or the shape. The correlation coefficients for the sample multivariate plot are shown numerically in each box of Figure 30, and also represented visually via the blue ellipses. The multivariate plot shows more than just the correlation. It also gives an idea as to the strength of the relationship. This is estimated as the slope of a line fit through the points. The strengths are shown visually in Figure 30 by the red lines. The thick blue outline corresponds to the portion of the multivariate plot typically represented qualitatively in the roof of the QFD.

One important note is that the surrogates are only valid within the ranges used to create them. This highlights an important consideration when comparing qualitative and quantitative estimates. It is possible for a relationship to be strong within a certain range, but weak within another range. As an example, Figure 31 shows a

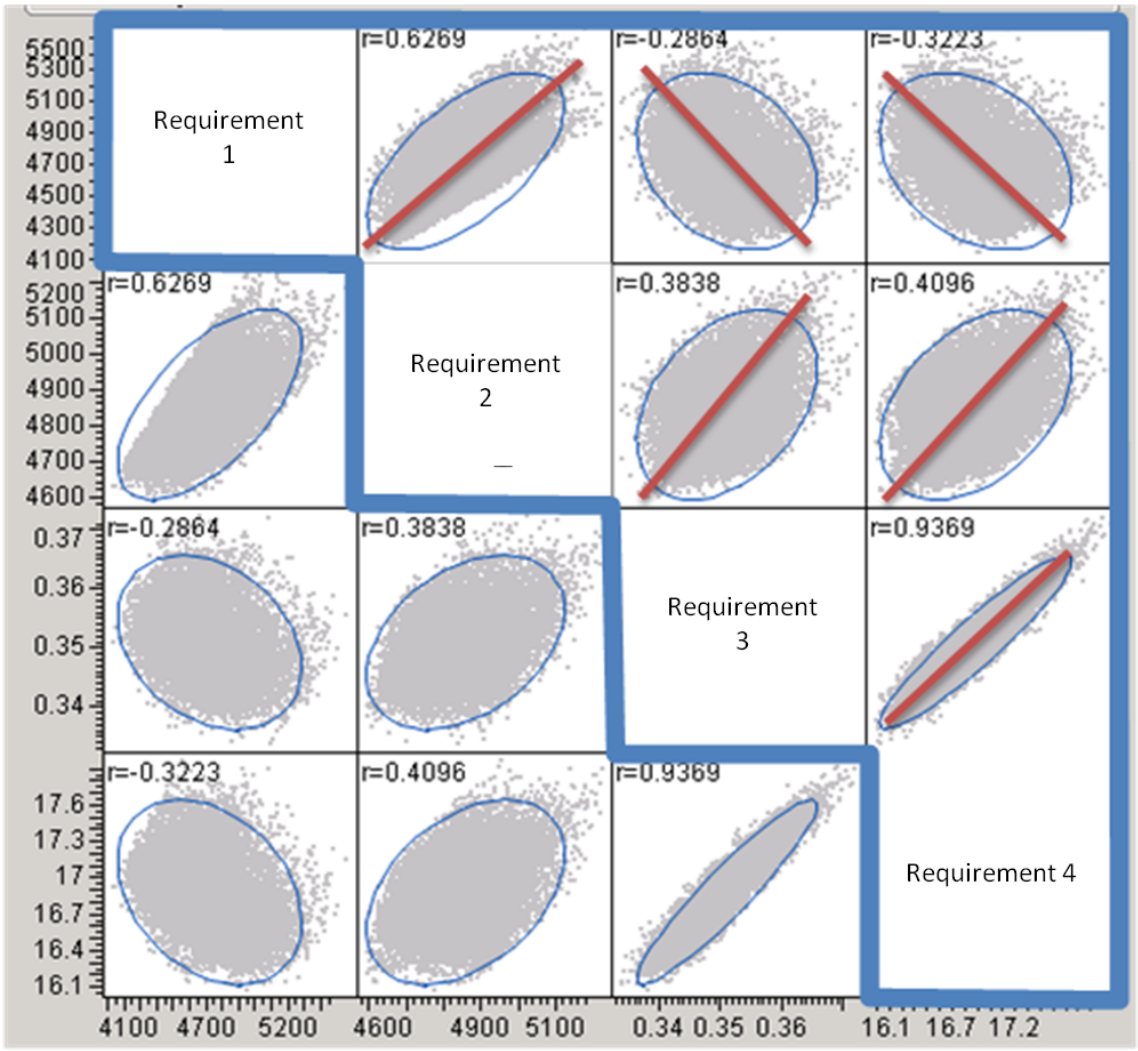


Figure 30: Example Multivariate Plot with Correlation Coefficient. Adapted from [118]

notional representation of the relationship between drag coefficient and Mach number is used. Although Mach number has an overall strong effect on the drag coefficient, it has little to no impact in the subsonic regime. If the SME used to elicit qualitative estimations is not clear on the assumptions of the problem at hand, the answer given regarding the magnitude of this relationship may vary depending on his tacit assumptions about the given problem, which would not be captured in the QFD. Furthermore, multiple SMEs may give very different answers if they have different tacit assumptions. This is an important consideration when conducting QFD, and

highlights the importance of clear and consistent assumptions.

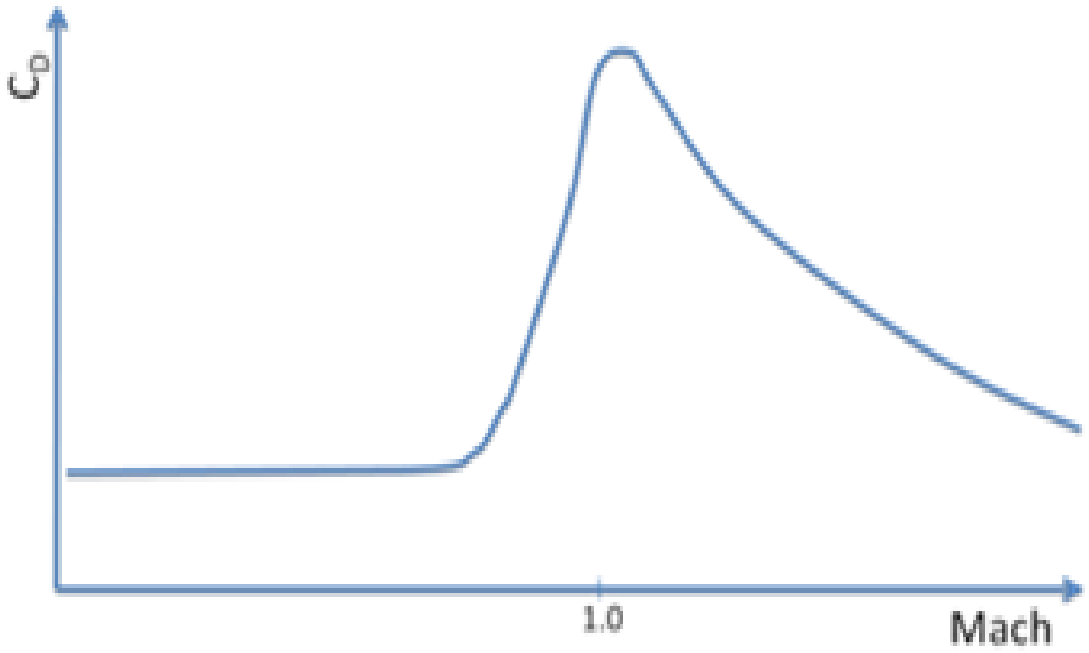


Figure 31: Notional Plot of CD vs Mach. Reproduced from [118]

The surrogate models are mathematical formulas representing the transformation between engineering variables and requirements, and the multivariate analysis provides the transformations between the requirements and themselves and the engineering variables and themselves. Figure 32 shows the analogy between the modeling and simulation results and the relations that are described in the QFD.

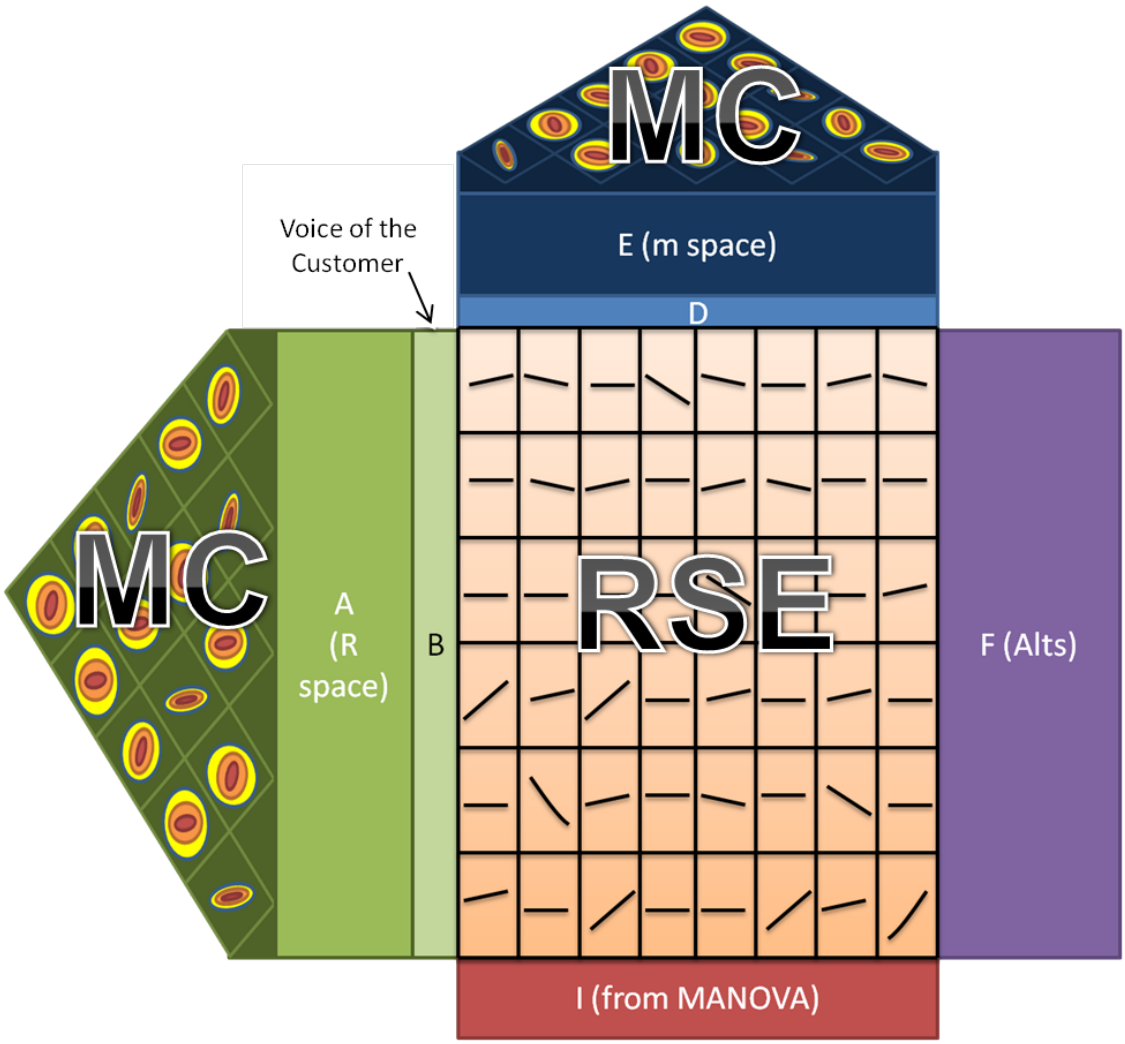


Figure 32: Analogy between M&S results and QFD. Reproduced from [118]

3.3.3 The ROSETTA Framework

To obtain a better understanding of the relationship between the QFD and Modeling and Simulation, it is helpful to better understand the mathematical nature of the relationships subjectively expressed by SMEs in the HOQ. An initial understanding of the mathematical foundations for ROSETTA is found in [118], and is expanded here. The transformation formulae described here are an extension of the model transformation rules presented in Dickerson and Mavris in [54, 55]. For the purpose

of this discussion, the requirements are referred to as members in the set of dependent variables $R_1, R_2, , R_n$, and the metrics are referred to as members in the set of independent variables $m_1, m_2, , m_p$.

In the HOQ body (the 'C' matrix in Figure 26), the SMEs are estimating the strength of the relationship between a particular r_i and a particular m_k. In mathematical terms, this mapping matrix can be described as the relational transformation between the r-space and the m-space where, for the (i,k) cell in the 'C' matrix, this transformation is represented by the slope of the partial derivative, $\frac{\delta Ri}{\delta mk}$. However, if one were to assert that this partial derivative fully and completely represents the sensitivity of R_i to m_k, it would then imply the assumptions that the $R_i s$ are completely independent of each other and that the $m_k s$ are completely independent of each other. If these assumptions are untrue, it follows that these partial derivatives are insufficient for capturing the total sensitivity of R_i to m_k. In real problems, it cannot be assumed that these assumptions will hold true. In fact, the existence of correlations in the greenhouse or roof imply mathematical dependence, and the existence of these relations imply that the assumption of independence does not hold.

To formalize the mathematical framework linking QFD and M&S, it is first necessary to establish and formalize an overall objective function. It is assumed that the goal of engineering analyses in design and technology tradeoff is to evaluate and compare potential solutions against a set of requirements. This implies that the framework being developed is solving a multi-attribute decision-making problem. In general, the value of any solution against all requirements is evaluated using an overall weighted function. This will be referred to as the total Quality Function, Q. This is similar to the approach used by QFD, which evaluates the importance of each metric based on its estimated impact on a set of weighted requirements. Stated mathematically, the QFD uses a linear Q, shown in Equation 3.

$$Q = \sum_{i=1}^{n} w_i R_i \tag{3}$$

Envisioning the values in the QFD as qualitative estimations of the strengths of linear slopes allows them to be envisioned as partial derivatives. This thinking provides a new perspective on the QFD analysis results. The relative importance scores calculated in part 'I' of the HOQ can then be seen as an approximation of the partial derivatives of Q with respect to each metric, where Q is weighted according to the customer weightings given in section 'B'. This equation is shown in Equation 4. This is only a partial derivative, however, because it does not account for the relationships between the metrics.

$$\frac{\delta Q}{\delta m_l} = \sum_{i=1}^{n} w_i \frac{\delta R_i}{\delta m_l} \tag{4}$$

The ROSE approach is not sensitive to the shape of the Q equation. Standard QFD makes the assumption that all metrics are linearly related to all requirements by using a constant value to describe the strengths of the relationships in the body of the QFD. The more generalized ROSE approach, however, allows these relationships to be any differentiable function. This means that if a subject matter expert is able to give the shape of the relationship (quadratic, exponential, etc), this can be captured in the metric rankings. The metric rankings would then become a function of the metric values, allowing decision-makers to understand the impact of the values of the metrics on the rankings of the metrics.

When performing modeling and simulations, there is a technique that allows the sensitivities of multiple responses to metrics or design variables to be calculated. This is called the Multivariate Analysis of Variance (MANOVA). MANOVA can be a particularly powerful analysis technique if it is known or suspected that the dependent variables are correlated [24]. MANOVA does not, however, necessarily give the sensitivity of Q to each metric, unless all of the regressors are independent. Instead, it

provides the relative criticality of each metric to correctly capturing Q in a model of the order of the highest regressor [9]. In this way it may be useful for eliminating redundant metrics. However, because RSEs or other forms of surrogates have been obtained that have potentially higher-order relationships, it is desirable to have a way of determining these derivatives that takes into account the effect of these higher-order relations. An approximation of these derivatives can be found numerically using the RSEs, or even the M&S environment itself if it is not too computationally intensive. ROSETTA offers an alternate approach to obtaining this information by thinking of the problem based on its set relations. If the relations between metrics are known (either from the physics, the modeling, or the SME estimates), and the RSEs which map the metrics to the requirements are known, then the total derivative of Q with respect to each metric can be found by applying Equation 5. It interesting to note that the addition of metric sensitivities allows for the total derivative of Q with respect to each metric to be calculated. Since this information is actually contained in the roof of the QFD when performing qualitative analyses, replacing the QFD math with this construct will allow for the metric relations to be taken into account, even in qualitative analysis.

$$\frac{dQ}{dm_l} = \sum_{i=1}^{n} \sum_{k=1}^{p} w_i \frac{\delta R_i}{\delta m_k} \frac{\delta m_k}{\delta m_l} \tag{5}$$

Additionally, it is not necessary that the total quality function have constant weightings on the requirements as suggested by the equation above. It is in fact possible that any differentiable quality function can be used, where the weightings are obtained by differentiating Q with respect to each requirement. This results in Equation 6.

$$\frac{dQ}{dm_l} = \sum_{i=1}^{n} \sum_{k=1}^{p} \frac{\delta Q}{\delta R_i} \frac{\delta R_i}{\delta m_k} \frac{\delta m_k}{\delta m_l} \tag{6}$$

The application of this equation gives a generic way to determine the sensitivities

of metrics to the full set of requirements, regardless of what information is available. This equation is insensitive as to the source of the data, so quantitative estimates can be used where modeling and simulation is available, and qualitative estimates can be used to fill in the gaps. Thus, decision makers are not restricted in information when models are not available, but instead, can begin with a qualitative approach and update the information over time as modeling and simulation results become available to decrease uncertainty. Using an analogous relational structure to the QFD, the data framework for this approach is shown in Figure 33, and can be used as a generic framework in which to track and store all data.

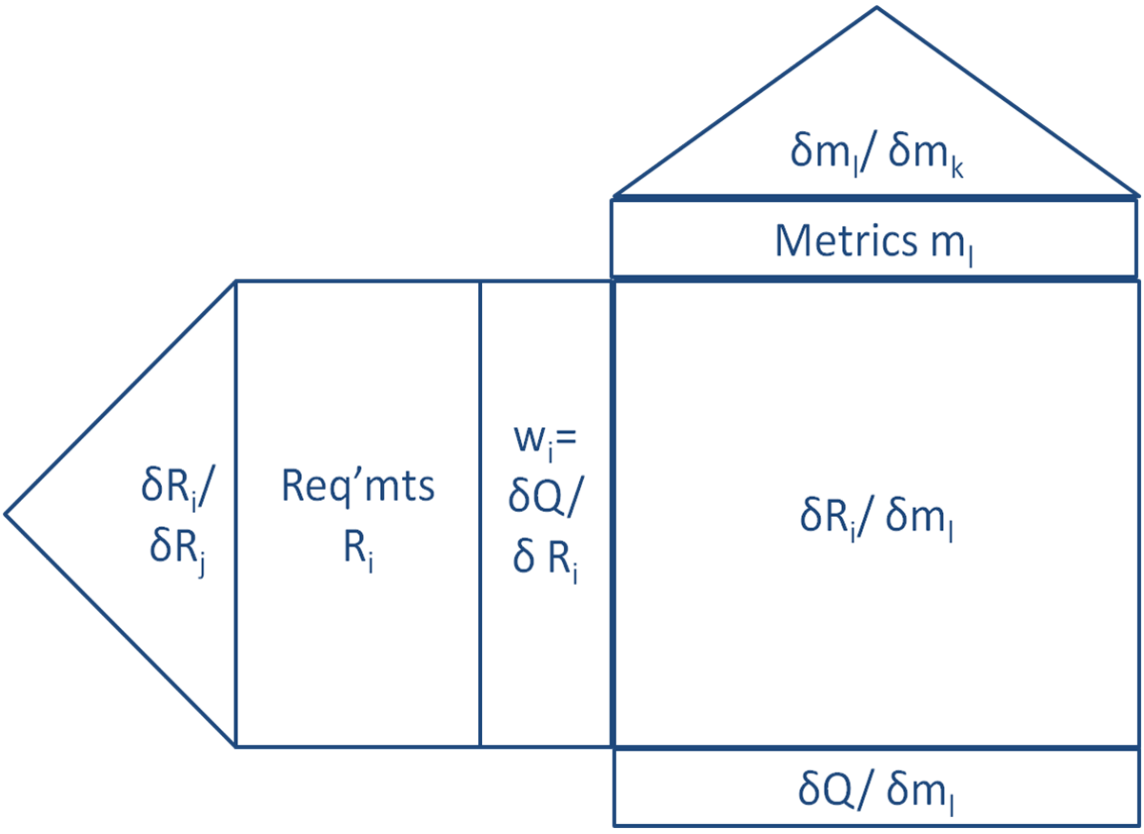


Figure 33: Generic Relational Structure for ROSETTA. Reproduced from [118]

The ROSETTA framework has multiple potential applications in the context of CBA. In the metrics derivation space, it provides a structured method for decomposing capability- and mission-level requirements into metrics, and verifying that the

set of metrics selected fully capture the capability space. Since initial qualitative mappings between the metrics and capabilities will be captured, it provides a way to determine which metrics most need to be improved to meet the most important requirements. If correlations between metrics are captured in the roof, these can be used to eliminate redundant metrics, and thus reduce the data gathering requirements. If ROSETTA is applied to project the relations between the metrics and the requirements to the requirements space, relationships between requirements can be captured and this can be used to eliminate redundant requirements (in which improvement of one guarantees improvement in another) and better focus the study. Furthermore, if the trends are estimated in the mappings between metrics and requirements, the performance values against metrics for the current baseline can be included in ROSETTA and projected to the requirements space to aid in performing gap analysis. Finally, the ROSETTA framework can act as a common framework for storing data and updating analysis results as more information becomes available. As it can mix the qualitative and quantitative data, it allows initial estimates on the relationships to be made by SMEs early on, and then to be updated over time ans more and more quantitative analysis is done. Furthermore, it allows for SME data to be used in places where quantitative models are not available in the given time frame, thus allowing for all things to be included in the study despite various fidelity data.

3.4 Existing Methods for Alternative Generation

3.4.1 Morphological Analysis

One challenge to decision making in the early phases of SE and design is the unfathomable number of alternative solutions available. In the design of a system, the number of alternatives can be calculated combinatorially by considering the number of components of that system and the number of possible of alternatives for each

Table 5: Matrix of Alternatives for Pencil Design

		Alt 1	*Alt 2*	*Alt 3*	*Alt 4*	*Alt 5*	*# Alternatives*
Lead	*Material*	Graphite	Solid Graphite	Charcoal	Carbon		4
	Grade	1	2	2.5	3	4	5
	Size	.5mm	.7mm				2
Casing	*Material*	Wood	Plastic	Paper			3
	Shape	Round	Rectangular	Hexagonal	Triangular		4
	Width	5mm	6mm	7mm			3
Eraser	*Type*	Fixed	Retractable				2
						Total	2880

component [184]. As a toy example, consider the design of a new pencil. The pencil has three components; the lead, the casing and, the eraser. Each of these components can then be subdivided into categories to describe the alternative space. In this example, between 2 and 5 alternatives are considered for each category. The matrix of alternatives for the pencil is shown in Table 5. By counting the number of alternatives in each row (done in the right hand column of the matrix) and multiplying these numbers, the total number of alternatives for this pencil design is found to be 2,880. A pencil is significantly less complicated that the systems typically considered in systems engineering processes. Ritchey [149] presents an example study from the Swedish National Defence Research Agency (FOI) considering the design of bomb shelters. The alternative space for this study considered 10 dimensions for alternatives, and contained over 500,000 possible bomb shelter configurations. In the example for a long range strike aircraft presented by Engler [63] and shown in Figure 34, there were roughly 125 trillion alternatives available for the design.

Enumeration of alternatives is often done using morphological analysis. General Morphological Analysis (GMA) was developed originally by Fritz Zwicky in the 1960s and has since been applied to a wide variety of applications to aid in the investigation of the relationship structure in large and complex problem spaces. In general, GMA is used to explore the full set of possible relationships in a given problem space. This is done by first describing the problem space through a set of n categories,

and enumerating all of the options for each category. Next, all of the options are set against each other in an n-dimensional Zwicky box, with each cell of the n-dimensional box representing one unique possible configuration. However, recognizing that not all of these cells will necessarily result in a feasible configuration, a pairwise comparison is done between each and every option to determine whether those two options are consistent (or feasible to select together). Zwickey refers to this as the *principle of contradiction and reduction*. Inconsistency can arise from a physical incompatibility, a judgment that this pairing is highly unlikely or impractical, or a decision by the user that this pairing should not be considered. From this consistency analysis, it is possible to reduce the number of solutions in the total space by removing those that violate the consistency checks [148, 184].

One derivative of GMA is the Interactive, Reconfigurable Matrix of Alternatives (IRMA), which can be used for visualizing and organizing the alternative trade space because it provides a clear and structured way to display and understand the many systems and where they fit in the overall functional decomposition. It is a useful tool for brainstorming of the solution space. It clearly displays all combinations, thereby showing combinations that might otherwise have gone un-investigated. A compatibility matrix in the background helps to capture incompatibility and some interoperability, and supports creation of the SV-1 and SV-2. It can help to capture tacit knowledge by means of the compatibility matrix and filters and it gives an appreciation for the size of the design space. It is a visually appealing and intuitive interpretation of the problem space. The IRMA can be used to create a physical decomposition, a functional decomposition, or some combination of the two.

The first step in developing an IRMA is to create a Matrix of Alternatives (MoA) which lays out alternatives for each category in a matrix, and fully describes the alternative space in this way. As an example, a system may be decomposed into categories based on its component-level needs. For example, an aircraft can be decomposed into

the wing, fuselage, tail, landing gear, and engine. Then, for each of these components, options are laid out. For example, the wing may be rectangular, delta-shaped, trapezoidal, swept, etc. The engine may be a turboprop, turbofan, turbojet, ramjet, etc. A more detailed decomposition can be done to increase the fidelity of the options space. Once the options are laid out, a compatibility matrix is created which lists all options against each other, and each is rated as compatible or incompatible. Options can be incompatible within the same category (for example it is impossible to have both a delta wing and a rectangular wing) or across categories (for example a rectangular wing and a ramjet is a nonsensical combination and thus incompatible). This is consistent with Zwicky's morphological analysis. However, the IRMA takes this one step further by using the information to create an interactive matrix, where all alternatives are laid out and each has a box where the user can select or reject each option. As selections are made in the matrix, incompatible options in this and other categories are automatically eliminated from the problem space based on invoking the relationships captured in the compatibility matrix. Furthermore, other dependencies can be captured as well. For example, if the selection of any one particular option forces the selection of another, this can be captured in the IRMA. [63, 54] An example IRMA interface presented by Engler is shown in Figure 34.

There are several gaps in this technique, however, when applied to large architecture problems. First, it can become extremely large for architecture based problems. Second, many aspects of the problem, such as the variations in operational concept or process sequencing, are very difficult to represent in this format. Third, it does not fully capture the interdependency inherent in complex systems. The compatibility between system elements is dependent on interface type and interoperability requirements, and this can not easily be captured with the morphological analysis or the IRMA. Finally, it does not include information about evolution of the system over time.

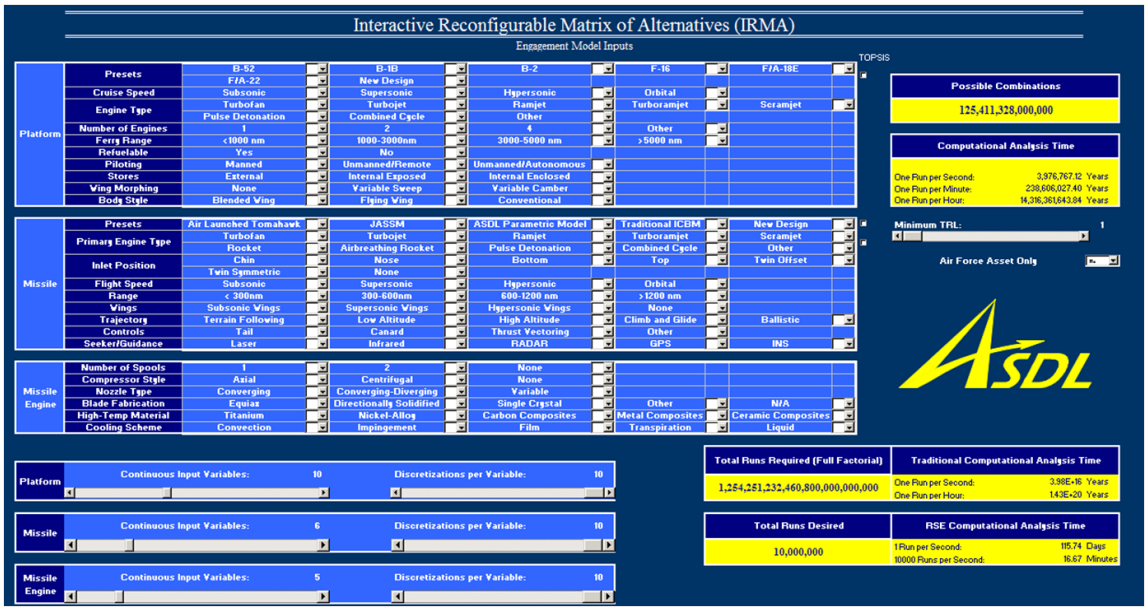


Figure 34: IRMA for Notional Long Range Strike Example, Developed by Engler [63]

3.4.2 The Alternative Space in System of Systems

The challenge of the sheer number of possible alternatives is compounded in SoS problems. In fact, not only is the number of alternatives extremely large, but the alternatives are varied in their type, including alternatives across all aspects of the DOTMLPF spectrum. Not only is it difficult to gather enough information early-on to make an informed decision, but it is difficult to even determine the criteria on which two extremely different solutions can be compared. In the case of some alternatives, like doctrine changes, just determining how to measure and quantify performance can be challenging. Even justifying the acquisition of a new system can be difficult, because it must be shown that the same level of mission cannot be achieved with a new arrangement or new uses of existing systems. In fact, even if there were only two alternatives posed for each of the DOTMLPF areas, and these options could be selected alone or in conjunction with other options, there would be 2^7 (128) alternatives. If there were three options per area instead of 2, there would be 3^7 (2,187) alternatives. Since in reality there are many more than two or three

options per area, it is not difficult to see that the alternative space is so large that there is no way that every combination of solution options could ever be explored. To further illustrate this challenge, consider a simple mission which is comprised of completing 10 activities. Then consider that these activities can be performed in two different sequences, thus creating two operational alternatives. Further, consider that each activity can be performed by one of three candidate systems. Next, consider three possible organizations that could be responsible for conducting this mission. Last, consider that there are two types of networks being considered for enabling communication in the architecture. There are then 2 organizational alternatives $\times$ 3^{10} system alternatives $\times$ 3 organizational alternatives $\times$ 2 network alternatives , resulting in a total of 708,588 alternatives.

Within an architecture framework, alternatives are represented by changes to one or more of the architecture products. There are many ways that the architecture products can be changed. Using a DoDAF perspective, operational changes can be made to the operational products of the architecture. This includes changes to the activities for which a node is responsible, the addition or subtraction of operational nodes, or changes to the information produced by or required by a node. In addition, changes can be made to the interfaces between nodes. This would include changes to the information flows or information exchanges. Changes can be made to the organizational responsibilities by changing the organizations responsible for particular nodes or by changing the overall organizational structure supporting the architecture. Changes can be made to the activities used to accomplish capabilities, as well as to the sequences in which activities are performed and the timing of those activity sequences. System changes are made within the system products. While many of the operational changes will induce system level changes, there are also alternatives that affect only system products. For example, the particular systems or services chosen to support the needs of operational nodes or interfaces can be varied. The way in

which information is exchanged between systems can be varied. The functions that are performed by systems to meet operational activity needs are also candidates for change, both in the choice of the functions themselves and which systems are tasked with performing those functions. The ordering and timing of functions can also be rearranged. The technical standards used in the implementation of the architecture can also be varied in the TV. It is important to realize that changes to one product may impact another product, and thus it is important to ensure that changes are represented across all products which are affected in order to maintain consistency.

Despite this large number of architectural alternatives, CBAs are still often limited to a very small number of alternatives in current practice. In fact, while the United States Air Force Early Systems Engineering Guidebook calls out the need to explore as many materiel and non-materiel solutions in CBA as possible, the guidance for downselection still focuses heavily on materiel solution criteria (such as technology maturity), and calls out that two more solutions be carried forward for analysis [167]. The DoDAF standard explicitly calls out an as-is and to-be architectures as the architectures that need to be considered when evolving an SoS, and does not consider the possibility of describing multiple architectures for analysis and consideration when exploration of future architecture concepts [50].

Thus there are several criteria for a design space exploration method for CBA. First, it must be able to capture and define the large number of architectural alternatives available for consideration during the early phases of acquisition and systems engineering. Next, it must have a way to filter through the design space and find only promising alternatives to evaluate, while eliminating those that are either not realistic or are not expected to meet mission goals. Finally, because even filtering processes will still leave large numbers of alternatives to be evaluated, there must be a way to quickly and accurately evaluate remaining alternatives. This means that the tools and techniques used for evaluation must be rapid while providing enough

detail to distinguish between alternatives, and that there must be a way to quickly generate inputs to these tools.

3.4.3 Discussion of Interoperability

Network centric warfare has become increasingly important in the defense community. In fact, network centric warfare is now the current doctrine of the US military [169, 170]. Network Centric Warfare (NCW) has been defined by Alberts et. al. [7] to be "an information superiority-enabled concept of operations that generates increased combat power by networking sensors, decision makers, and shooters to achieve shared awareness, increased speed of command, higher tempo of operations, greater lethality, increased survivability, and a degree of self synchronization." In a report to Congress, O'Rourke describes the focus of network centric warfare as "using advanced information technology (IT), computers, high-speed data links, and networking software, to link military personnel, platforms, and formations into highly integrated local and wide-area networks" [137]. While it is generally assumed that increased net-centricity means better combat performance and efficiency, there is a cost associated with increased interoperability. However, not all experts are convinced that net-centricity leads to improved warfare [105]. A recent report expressed four areas of concern with respect to the ideas of network centric warfare, which are (1) physical attacks on critical information nodes; (2) electromagnetic attacks against ground, airborne, or space-based information assets; (3) cyber attacks against information systems; and (4) attacks and system failures made possible by the increased level of complexity inherent in the multiplicity of advanced systems [68].

Furthermore, network centric warfare, by its very definition, will necessarily cause an increase in the complexity of the SoS, according to the definition of complex systems presented in 1.1. The definition specifies that characteristics of complex systems include rich interactions among elements (which is a critical enabler for network

centric warfare), a loosely organized interaction among elements, being open to the transfer of information across the system boundary, and being diverse in technology, operation, geography, and conceptual frame, all of which necessarily apply to network centric warfare. Although added complexity can result in increased performance, it does typically come at a cost [84]. In NASA's recent decadal survey, it was shown that while increased complexity did not have a strong impact on the probability of program success, it did have a strong impact on the program cost, as shown in Figure 35. Thus, it is clear that trade studies need to be performed early in the acquisition process to clearly understand the advantages and costs of implementing network centric concepts and programs in the military SoS.

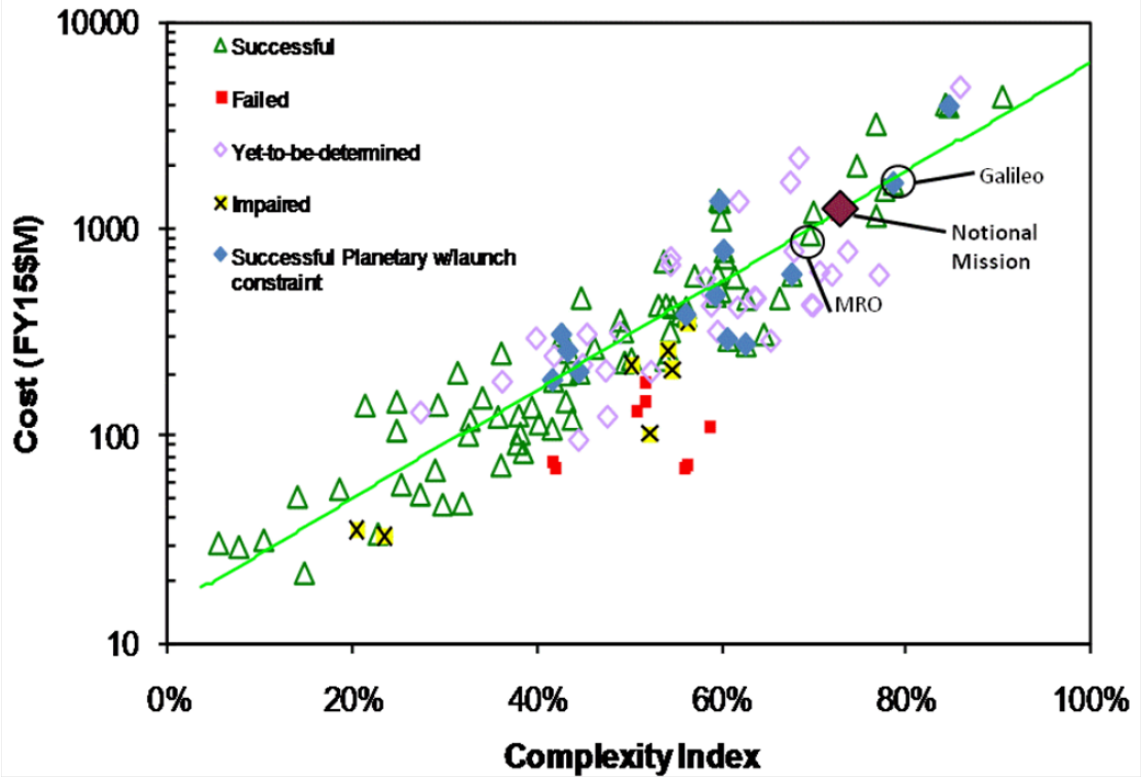


Figure 35: Relationship Between Complexity and Cost Using NASA's CoBRA Model. Reproduced from [36]

The shift from platform-centric warfare to network-centric warfare requires a significantly increased emphasis on the interfaces of the SoS architecture, as the interfaces enable the information sharing and collaboration that fundamentally underlies network centric warfare. When considering network or interface-based alternatives in an SoS, there is somewhat of a chicken and egg problem. The performance of the physical architecture in the operations described in the operational architecture products is dependent on the information flow specified in those products being enabled. In order to understand the requirements of the system interfaces, information about the needlines and the systems residing in the nodes on either end of the needlines is required. However, in order to determine whether the systems resident at the nodes are feasible with respect to enabling the operational architecture, the interface requirements must be known. Furthermore, since the transfer of the needed information can happen any number of ways with varying degrees of accuracy and precision, there are a large number of alternatives associated with the implementation of the interfaces. Ultimately, there is an information systems architecture operating in conjunction with the physical SoS architecture that together enable completion of the mission. It has been observed that most SoS analyses assume that the network architecture is in place and will allow the desired communication to be effected, while most network architecture analyses assume fixed systems operating under a set of known interface standards with rigidly controlled interfaces. However, the interaction between changes to the physical architecture and changes to the network architecture is rarely studied. Furthermore, most network architecture analysis techniques require a significant amount of information about systems that may not be available during CBA, particularly when considering new systems that have not been fully designed yet. As the DoD pushes toward a net-centric warfare paradigm, interface requirements that enable increased levels of interoperability will become a driving factor in SoS design and implementation, and therefore must be considered during the CBA.

In a military context interoperability is defined as [102]:

1. The ability to operate in synergy in the execution of assigned tasks. 2. The condition achieved among communications-electronics systems or items of communications-electronics equipment when information or services can be exchanged directly and satisfactorily between them and/or users. The degree of interoperability should be defined when referring to specific cases.

Dickerson goes on to say "As noted previously, in a complex system, it is the interfaces and connections that provide unique, value-added FoS [Family of Systems] functions" and that interoperability has been shown to be a force multiplier where systems already offer inherent mission capability [53]. Thus, the movement and sharing of information between systems is a key architectural consideration that impacts mission effectiveness.

There are several standards for describing IOL, with Levels of Information Systems Interoperability (LISI) being the most commonly used. LISI was developed by the C4ISR Architecture Working Group to compliment DoDAF and to work with JCIDS. LISI has five interoperability levels measured across four attributes, and is used to measure the interoperability between two systems. The four attributes used to describe interoperability are Procedures, Application, Infrastructure, and Data (PAID).A summary of the LISI framework can be found in Figure 36 [28].

Description	*Computing Environment*	*Level*	P	A	I	D
Enterprise	Universal	4	Enterprise Level	Interactive	Multi-Dimensional Topologies	Enterprise Model
Domain	Integrated	3	Domain Level	Groupware	World-wide Network	Domain Model
Functional	Distributed	2	Program Level	Desktop Automation	Local Networks	Program Model
Connected	Peer-to-Peer	1	Local/Site Level	Standard System Drivers	Simple Connection	Local
Isolated	Manual	0	Access Control	N/A	Independant	Private

Figure 36: A Summary of the LISI Model. Reproduced from [28]

3.5 Existing Techniques for Quantitative Alternative Analysis of SoS

3.5.1 Quantitative Modeling Techniques

This section will present several types of modeling that have been used in the literature to model different aspects of SoS. However, in order to assess the applicability of each of these models to this research effort, a framework for describing and comparing models is needed. As was discussed previously, there are several requirements for models to be used for executable architecting, most particularly, the ease of automation. The fact that the goal of this research is to use these models during CBA presents a further set of criteria, particularly a short execution time for the model and the need only for information inputs that are expected to be available during CBA. It is also expected that understanding the level of fidelity that can be provided from the model is important so that the level of certainty in the results can be understood. However, it is difficult to assess the fidelity of a modeling type, since the fidelity of the model is dependent on how much information is available to be included and

the accuracy of that information. Thus models can be only assessed based on the fidelity level they are capable of achieving, rather than the fidelity level they would necessarily achieve in any given application of the technique.

In addition to these criteria, the models also must be able to represent the problem at hand, and must be capable of producing the required metrics from the metrics derivation step. Models can be qualitative or quantitative depending on what type of information is available. In order help determine if the models can capture necessary aspects of the problem, quantitative models will be described according to several general characteristics. First, models can be characterized based on whether they are stochastic or deterministic. Second, models can be either static or dynamic, in that a modeling type can either capture behavior at a single point in time or capture behavior over time. Last, dynamic models can be either continuous or discrete in the way that time is handled within the model. This characterization of models leads to the model characterization tree shown in Figure 37.

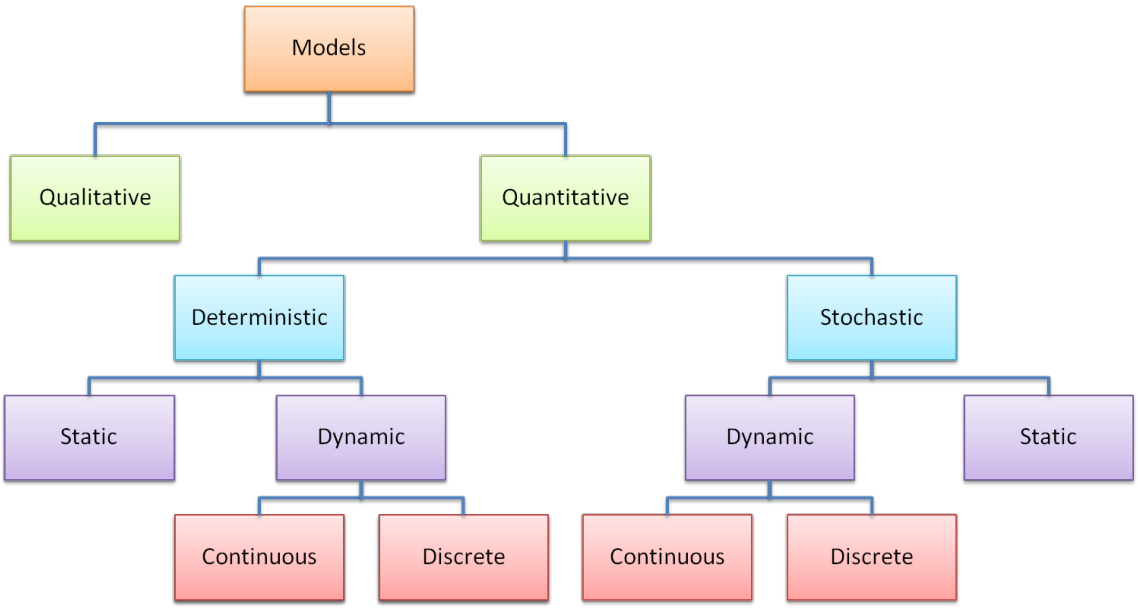


Figure 37: Model Characterization Tree

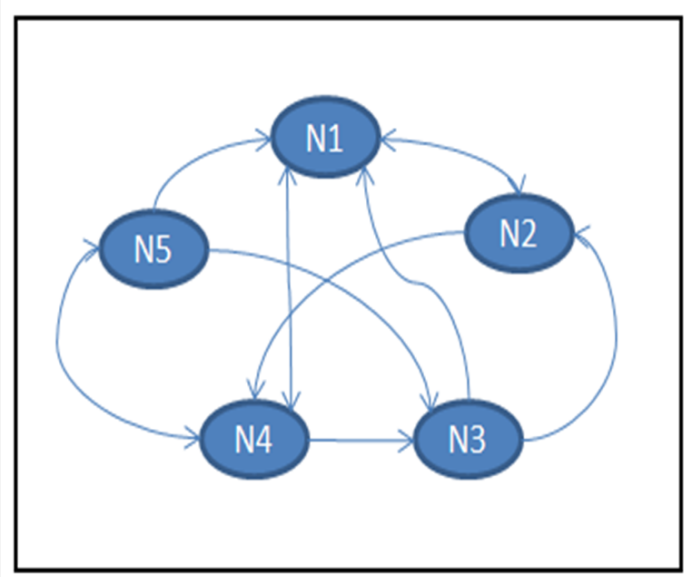
As was previously discussed, it is desirable to use quantitative modeling as much as possible. Furthermore, SoS are generally stochastic in nature, due to the inclusion

of humans as part of the system as well as the large number of interactions with the environment and between elements in the SoS. It is also very likely in SoS problems that it will be important to examine the dynamic behavior of the SoS over time. Thus, the most applicable models for capturing SoS behavior are likely to be those which are stochastic and dynamic, although models within other branches of the tree may be sufficient for capturing certain aspects of the SoS. With these considerations in mind, the following sections will discuss several types of models commonly used in SoS which are candidates for inclusion in this research. A discussion of these modeling types and how they can relate to DoDAF was previously presented by the author in [76], and the discussion in the following sections is adapted from this work.

3.5.1.1 Graphs and Network Theory

A mathematical graph consists of a set of vertices (or nodes) and a set of edges (or arcs). There is a variety of types of graphs, the simplest of which shows only a set of vertices connected by a set of edges. These edges have no properties associated with them, such as a direction or weight. Adding a directional flow to an edge makes a directed graph. Edges or nodes can also be weighted, by overlaying additional information onto the graph. This information might include node or edge capacity or edge length. Overlaying capacity might allow the flow of information through a computer network to be modeled, and overlaying length may allow a network of roads to be modeled. Using this information, algorithms exist to find the maximum capacity of the network, or the shortest path through the roads. Cyclic graphs are graphs which include cycles. Cycles are paths through a graph that return to the node from which they initiated; where graphs with no cycles are known as acyclic graphs. Graphs can be represented by using either a pictorial form, or by using an matrix form known as an adjacency matrix. Both of these forms of representation are mathematically equivalent. Other forms of graph representation will not be discussed

here. The pictorial form is easy to understand and interpret; however, the adjacency matrix form is more useful for computer-based analyses [182]. Figure 38 shows an example of a graph represented using both the pictorial version and the adjacency matrix. The adjacency matrix is read as follows: there is an edge from the node in a row to the node in the column if there is a '1' in (row,column) location in the matrix. An undirected graph has a symmetric adjacency matrix. If values besides '1' are used, these numbers are used to represent weightings placed on the edges of the graph. These weightings can represent any number of physical phenomena (such as capacity or length as discussed above), and will be discussed more in the examples provided [182].



	N1	N2	N3	N4	N5
N1	0	1	0	1	0
N2	1	0	0	1	0
N3	1	1	0	0	0
N4	1	0	1	0	1
N5	1	0	1	1	0

Figure 38: Pictorial Representation of Graph and Associated Adjacency Matrix [76]

As an example of how graphs can be useful for modeling real world phenomena, imagine that a graph is made to represent a communications network. This would similar to a combined SV-1/SV-2 DoDAF model. The information in the SV-1/SV-2 would include nodes and edges of the network, as well as what systems are being used across the edges, thus providing information about the bandwidth or flow capacity of each edge. Depending on the type of communications network being depicted, there

are several types of information that can be obtained by representing this architecture view as a graph and performing analysis on the information contained within it. For example, if the communications network is focused on transporting information between several separate physical locations or along several choices of paths, graph theory can be used to determine the shortest path, determine the maximum flow capacity through the network, and identify where bottlenecks are expected to occur. This analysis is dependent on having identified sources and sinks in the network and at least one viable path between them [182].

If the network is cyclic, (i.e., the information or resources must return eventually to where they began), graph theory can be used to gain further insight into the communication network. From spectral graph theory, there is a measure called the Perron-Frobenius Eigenvector (PFE). The PFE gives an indication of how much each element contributes to the autocatalytic cycles in the graph. Stated another way, the PFE helps to measure how central each element is to the success of the communication cycle. The PFE is the Eigenvector associated with the largest, real, non-negative Eigenvalue of the adjacency matrix. The Eigenvalue itself provides a measure of the networked effects by providing a measure of the number of autocatalytic sets in a network model [95]. Balestrini has demonstrated the use of PFE for analyzing the kill chain in [14].

To better understand the Eigenvalue, Figure 39 is adapted from Cares [29], showing how the structure of a network affects the PFE. The graph represents a simple combat network with four players, including a sensor (denoted by S) that detects a target, a command and control node (denoted by C2) that makes a decision to engage the target, an engagement node (denoted by E) that engages the target, and the target node (denoted by T). This very simple kill chain must be completed to be successful in a mission. In order to complete the kill chain, the target must be detected by the sensor, which then relays the target location information to command

and control, which then gives the orders to engage the target, after which the target is engaged.

The figure shows several examples of how this kill chain could be accomplished, and the PFE eigenvalue associated with each. In example a, there is no engagement node, and thus no cycle, meaning that there is no way to complete the kill chain. In this case, the Eigenvalue is zero, meaning there are no networked effects. In example b, the engagement node is included, representing the simplest possible functional cycle to complete the kill chain. This results in an Eigenvalue of one. Example c adds a second sensor, thereby increasing the chances to succeed, since now either sensor can successfully detect the target and begin the kill chain. Because the second cycle is enabled by the addition of a second sensor, the ability of the network to succeed is increased without requiring that all of the network elements be duplicated. This causes an increase the eigenvector to a value greater than one. It should be noted that the maximum value that the eigenvalue can achieve is equal to the number of nodes in the network. This example shows how the eigenvalue associated with the PFE can be a useful tool for comparing network structures. In this research, it is assumed that increased networked effects may be desirable, but that increases in networked effects are limited by cost and resource concerns [29]. One of the objectives of this work is to demonstrate when increased network effects are worth the cost and resource invest required to achieve them.

The PFE, as stated above, is the Eigenvector corresponding to the above Eignenvalue, and gives the centrality of the nodes themselves. Each component of the PFE corresponds to a particular node (in the order they are represented in the adjacency matrix), and when the PFE is normalized, these values give the centrality percentage of that node to the overall network. The highest centralities will correspond to those nodes which are included in the largest number of cycles [23]. This information can be very valuable, as it can identify points of vulnerability in friendly networks. If the

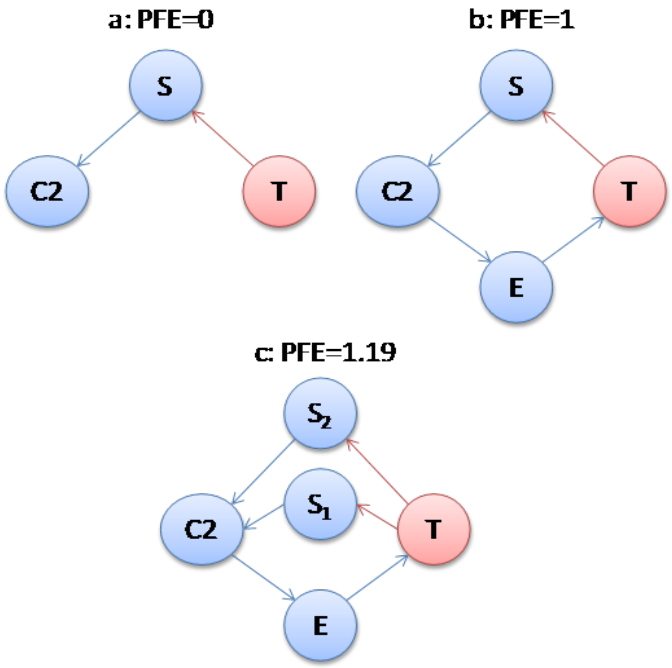


Figure 39: Influence of Network Structure on the Eigenvalue associated with the PFE. Redrawn from [29]

information is available to create the model for an enemy network, it can also be used to identify high priority targets will which best disrupt the enemy's communications.

A related value of interest, which is derived from the Eigenvalue, is called the Coefficient of Networked Effects (CNE). The CNE is essentially a measure of efficiency. It is calculated as the ratio between the Eigenvalue and the number of nodes in the network. This is useful because it normalizes out the number of nodes in a network, thus giving a way to compare networked effects across networks with differing numbers of nodes. The CNE obtains values between zero and one [29]. A method for developing and evaluating these parameters in the context of a military SoS has been presented by Balestrini [14], and details of the implementation can be found in his document.

As was observed before, the SV-1/SV-2 can be used to create a network model. Since this view focuses on systems resident in nodes and systems that enable the needed communications between them, it would be useful to also be able to analyze this view to determine how well the network meets its goals. Treating this view as

a graph would allow the identification of potential bottlenecks when the system is operating at full capacity, the maximum flow of information enabled by the network, and the relative criticality of each node to enabling this information flow. It could also give a measure of the redundancies which exist in the network. The use of mathematical graphs as commonly applied to analyzing networks (sometimes referred to as "network theory") provides algorithms to estimate all of these parameters, as was discussed above. The use of a SV-1 combined with a SV-2 to generate a Network Model for analysis of centrality has been previously shown by Griendling and Balestrini [77].

3.5.1.2 Markov Chains

Markov Chains are named after AA Markov, who first began studying them in 1907. A Markov Chain is defined as a mathematical model consisting of a set of states $S = \{s_1, s_2, \cdots s_m\}$ where the probability of transitioning between s_i and s_j at any given time step is p_{ij}, and p_{ij} does is independent of what states have been visited prior to the current time step. Stated in another way, the future state in a Markov Chain is dependent on only the present state and not the past states. Mathematically, this property is expressed for a discrete time Markov Chain in Equation 7, where X_n is the value of the random variable at time n [110].

$$P(X_{n+1} = i_{n+1}) \mid X_n = i_n, X_{n-1} = i_{n-1}, , X_0 = i_0) = P(X_{n+1} = i_(n+1 \mid X_n = i_n) \tag{7}$$

The transition probabilities are the probabilities of transitioning from one state to another, and are denoted as p_{ij}. These and are typically represented in a transition matrix. In the transition matrix, the columns reflect the current state and the rows reflect the state at the next time step. If probability of being in any current (initial) state is known, calculating the probability of being in any other state after n amount

of time is a simple prospect. This calculation is performed using Equation 8, where Φ is the vector of probabilities of being in a given state and P is the transition matrix containing the probabilities of transitioning from any state to any other state in a single time step [110, 78].

$$\Phi_n = \Phi_0 P^n \tag{8}$$

The limiting behavior of the Markov chain can also be examined, using Equation 9 [110].

$$\lim_{n\to\infty} \Phi_n = \lim_{n\to\infty} \Phi_0 P^n \tag{9}$$

There are a number of cases in which the limiting behavior of the Markov Chain is independent of the initial probability vector. In order for this independence to occur, the largest Eigenvalue of the transition matrix must be a simple Eigenvalue with a value of one. Additionally, the Eigenvector associated with this Eigenvalue must have all positive entries. If both of these conditions hold true, converting this Eigenvector to a probability vector will provide the long term stationary distribution of the Markov Chain. This is shown in Equation 10, where π is the stationary probability vector [110].

$$\lim_{n\to\infty} \Phi_0 P^n = \pi \leftrightarrow \pi P = \pi \tag{10}$$

Markov chain models have been applied to a range of problems, including the study of queuing theory [123] and the analysis of baseball [141]. Markov Chains can be modeled as discrete models, called Discrete Time Markov Chains (DTMC) (corresponding to the equations expressed above), or as continuous time models, Continuous Time Markov Chains (CTMC). In CTMC, the formulation is slightly difference, where transition probabilities are replaced by the use of transition rates.

These transition rates are, by definition, exponentially distributed, meaning that they exhibit the memoryless property. The memoryless property goes hand-in-hand with the definition of the Markov Cahin, as it means that a current sample from the distribution is entirely independent of (i.e. has no memory of) earlier samples. Like in DTMC, the stationary distribution can be studied in CTMC, and is the steady state behavior that would be observed if the simulation was run over an infinite time [140]. Markov chains are a natural fit for studying DoDAF models. Markov Chains can be used to study the dynamic state-space behaviors of the system, represented in the OV-6c and SV-10c products. The activities in these models represent the states, and rates of transition are given by the time steps in the product. Markov Chains have the advantages of a rapid run time and a simplified model that requires a relatively small amount of information, making them appropriate for early phases of design and acquisition when little information is available.

3.5.1.3 Discrete Event Simulation and Petri Nets

Discrete Event Simulation (DES) is a modeling technique which uses numerical analysis to analyze systems where the state variable(s) change only at discrete points in time [15]. Typically, a DES is comprised of servers and entities, in which the servers perform actions to process the entities. The servers have a maximum capacity of entities that can be processed in parallel, and a distribution of time to process an entity. DES has several paradigms in which it can be preformed: activity-oriented, event-oriented, and process-oriented. The activity-oriented paradigm breaks time into very small segments, and, at each time step, checks the progress of all activities. This can be very slow to execute, and simulations may take days to run, which is undesirable for early phase acquisition. The event-oriented paradigm solves this issue, and speeds up simulation by skipping over time steps in which no changes occur. To do this, a set of pending events is stored. Then, for each of these events, the times at which

they will occur are checked, and the simulation skips to the earliest time step in the group. This paradigm has the desirable characteristics for early-phase acquisition of being easy to implement, relatively rapid to execute, and flexible. The process-oriented paradigm models processes within the system as independent threads. The process-oriented paradigm has the benefit of a more modular code, thus making it a popular alternative for DES implementation [117]. Regardless of the method chosen for implementation, DES can be a useful modeling tool for modeling queues (which may be seen in logistics analysis or missions involving the movement of information or resources) and other similar processes.

One common implementation of DES is the Stochastic Petri Net (SPN). According to Hass and Shelder [79], "In the context of discrete event simulation, the marking of a SPN corresponds to the state of the underlying stochastic process of the simulation and the firing of a transition corresponds to the occurrence of an event." They further state that "for any (finite or) countable state GSMP (generalized semi-Markov process) there exists an SPN having a marking process that mimics the GSMP in the sense that the two processes (and their underlying general state-space Markov chains) have the same finite dimensional distributions" [79]. They conclude that "SPN's with timed and immediate transitions provide a general framework for discrete event simulation"[79]. This implies that in situations where a Markov Chain is unable to adequately describe the behavior of a process, a Stochastic Petri Net (SPN) may be an appropriate alternative framework for the implementation of a DES. Thus, the SPN gives a more generalized model for capturing the types of behavior typically measured using a Markov chain.

A Petri Net can be defined as a directed bipartite graph consisting of states (or locations), transitions, and directed arcs connecting states and transitions. Tokens are placed within the states, and may move between states according to a predefined set of transition rules. Petri nets are a stochastic modeling tool, which means that

repeated executions of the same set of inputs are not guaranteed to result in the same set of outputs due to the potential for multiple transition pathways between states [178]. An extension of the general Petri net is the stochastic Petri net, in which transitions are timed according to some probability distribution rather than a single value, which results in the addition of uncertainty to the transition times. This makes the SPN a particularly useful for accounting for uncertainty [120]. In fact, when examining communications networks, accounting for uncertainties in transmission times is of particular interest. Unlike most network models, which examine the maximum rates at which information can be moved, the SPN will look at the range of rates, accounting for the many factors that could slow down or delay the transmission. As an example of a case in which this may be important, consider a kill chain in which there must be a human-in-the-loop to give engagement orders. These orders are given on the basis of available data, which must be analyzed and interpreted, a process which can take varying amounts of time depending on the person performing the analysis and the tools available to them. It could further vary depending on the quality and quantity of available data. Another example would be a situation in which communication pathways are bogged down, and thus it would take longer than expected to send information through the network. Creating a stochastic Petri net model of the network and performing a Monte Carlo simulation on that model can help account for these types of uncertainties, as was demonstrated in [13].

The DoDAF documentation actually suggests the use of a Petri Net for developing OV-6 models [45]. Because the OV-6 models capture states and transitions between those states, they are well suited to the development of a Petri net model. In cases in which the architecture is being developed for time-critical missions, there is particular value in the use of a Petri net model to assess the impact of uncertainties on the mission completion time and identify key risk areas. Rather than simply estimating the amount of time it would take to execute a mission under normal or average

conditions, the Petri net could instead examine the range of conditions, and return the probability that the mission will be completed in a given amount of time.

Besides the SPN, there have been a number of other extensions made to Petri net models. While a full discussion of all the possible extensions is outside the scope of this thesis, there are two additional extensions of particular relevance that will be discussed here. The first extension of interest is the colored Petri Net. The colored Petri net allows the token to be differentiated or flagged using a color scheme. The colors can be used to either track specific tokens, or to denote tokens which have passed through a specific gate or exhibited a certain behavior (e.g. returning to the same state three times, etc). As an example, imagine that tokens represent pieces of information moving through a network. The colors could then be used to differentiate the types of information, or could be used to denote the addition of additional metadata to that piece of information. Color policies could be used to examine the impact on mission success of giving different priorities to different types of information. It would also be possible to create the model such that different types of information follow different behavioral policies in the model, such that there could be multiple information paths with multiple information types being represented at once. The use of colors can reduce the size of the model and the number of states required to create a model, thus reducing computational loads [179]. An example using a colored Petri net to model a distributed intelligence system has been developed by Levis in [111] and provides an example of the use of colored Petri nets for analysis of alternative architectures.

A second extension of interest is the application of age to tokens, called aging tokens. In this extension, the age of the tokens is tracked as the tokens move through the system. Events can be programmed to change the way in which the tokens age (e.g. a repair event may make a token's age younger). Furthermore, event transitions can be dependent on the token age, only allowing tokens that are greater or less

than a specific age to pass. Volovoi [178] has demonstrated the use of aging tokens in safety and reliability studies, as systems are often more likely to fail as they get older. The aging tokens can also be used to look at maintenance and repair policies. For example, a repaired system may have a shorter time between failures than a new system [178]. In the context of military communication systems, aging tokens might represent the age of a specific piece of information. As some information is only useful for a limited period of time, this would allow the impact of time to relay information on mission success to be studied. For example, the location of a moving target would only be useful until that target moved, and thus the information would expire if not acted on quickly enough. Tokens that exceed this age limit could be sent to a failed state, allowing a user to study the likelihood (or impact) of critical information expiring prior to being used. This enables trades in communications architectures to be coupled with mission execution. If too much information is being lost, it may suggest that bandwidths need to be increased, that processing times need to be shortened, or that alternate network architectures should be considered.

These extensions do not have to be used independently. For example, in a time-critical problem, the color of the tokens could be used to flag the expiration rate of the information, and the age could be used to determine whether the information has exceeded its expiration time. Alternately, colors can be used to denote the importance of a piece of information, and the aging can be used to examine how old the information is on average when it is reaches the point of being used.

The use of DoDAF products to develop executable architectures using Petri Nets has been demonstrated by AbuShaekh, Kansal, and Levis [2] using an air interdiction example. This work cites several challenges in using DoDAF to create executable architectures, namely a lack of standardization of products and a difficulty in making products computer-readable. It is also noted in this work that additional information will be required beyond that which is typically included in the DoDAF product

models. When the Petri net is based on the OVs, a set of initial conditions must be included to be able to generate the model. When the Petri net is based on the SVs, initial conditions, scenarios that characterize changes in the performance parameters over time, and an additional set of rules that addresses concurrency and asynchronicity must be included [2]. These elements are beyond what is required by the DoDAF specification, but are required overlays to the products in the context the creation of an executable environment. In order to use Petri nets to create a generalized executable architecture environment, these challenges will have to be addressed.

3.5.1.4 System Dynamics

System Dynamics (SD) is modeling and simulation technique developed in the 1950s by Professor Jay W. Forrester of the Massachusetts Institute of Technology to aid in the understanding of the dynamic behavior of complex systems and processes [161]. System Dynamics models use stock and flow diagrams and causal loops. Accumulations are represented using stocks, and the flows represent movement between stocks. May different types of things, such as money, materials, people, or equipment, can be represented in an SD model model. [109]. In the context of information systems, stocks will represent accumulation of data or information and flows will describe the flow of that data or information. The last element of a system dynamics model is the information links, which occur when a flow or a stock is influenced by another variable. The information links help to capture the dependencies between elements and variables, and create equations governing the dynamic behaviors of the system [109]. The ability of SD models to capture cause and effect chains make them particularly useful in capturing the dynamic behaviors of systems with feedbacks where systems cannot be accurately studied independently [158].

Close examination of SD models vs DES reveals that the two are actually very similar, and can be used to yield similar results. There are two key differences between

the two, however, and these differences will help determine which is most appropriate and when. First, SD are continuous and DES models are discrete. Second, and more importantly, is that SD models are deterministic, which DES models, as discussed previously, can be formulated as stochastic models [160]. As such, DES relies on the existence of historical data on the systems or on similar systems to estimate distributions associated with state transitions, but have the advantage of capturing a range of performances and the probability of behavior falling in different parts of the range. In contrast, values used in SD models usually represent average or expected performance.

Because the SD model is based on sources, sinks, and flows, it fits naturally with the DoDAF SV-4 (System Functionality Description). The SV-4 provides the flow of data between system functions, including the sources (creators) and sinks (users) of that data. It does require a more sophisticated set of inputs than the Markov Chains or DES discussed earlier, and thus may only be used if and when the required input information is available.

3.5.1.5 Constructive Simulation

The DoD M&S Glossary [42] provides the following definition of a constructive simulation, based on the distinction between it and live and virtual simulations:

> Live, Virtual, and Constructive Simulation: A broadly used taxonomy for classifying simulation types. The categorization of simulation into live, virtual, and constructive is problematic, because there is no clear division between these categories. The degree of human participation in the simulation is infinitely variable, as is the degree of equipment realism. This categorization of simulations also suffers by excluding a category for simulated people working real equipment (e.g., smart vehicles). Live Simulation: A simulation involving real people operating real systems.

> Virtual Simulation: A simulation involving real people operating simulated systems. Virtual simulations inject human-in-the-loop in a central role by exercising motor control skills (e.g., flying an airplane), decision skills (e.g., committing fire control resources to action), or communication skills (e.g., as members of a C4I team). Constructive Model or Simulation: Models and simulations that involve simulated people operating simulated systems. Real people stimulate (make inputs) to such simulations, but are not involved in determining the outcomes.

In general, constructive simulations are used within the DoD to simulate wargames with virtual actors and virtual equipment. This is a far less costly approach than using real people and real equipment, although there is some debate as to the accuracy of the results. However, the cost and time required to perform live or virtual simulations excludes these simulations from being used when one wishes to explore large areas of the problem space. Thus, in the early phases of acquisition, a constructive simulation approach can be used to explore many aspects of the problem. Although the results may not be as accurate as those from a live simulation, the constructive simulation can help identify important trends, examine emergent behaviors, and provide insight into the relative pros and cons of various choices. In fact, the following list has been provided by Chi, et al. [34], and shows when the use of constructive simulations is relevant for wargaming:

- Need emergent tactics/strategies
- Currently only suboptimal maybe possible depending on the [gamers'] skill
- Need faster speed than real time performance
- No synergistic interaction
- Need very fast decision

- Need a Robust repeatability
- Might make error by human intervention
- Increasing complexity in Decision-making
- Increasing importance of human factor in warfare

Agent-based models (ABM) are a type of constructive simulation. Bonabeau provides the following description of an agent-based model [22]:

> In agent-based modeling (ABM), a system is modeled as a collection of autonomous decision-making entities called agents. Each agent individually assesses its situation and makes decisions on the basis of a set of rules. Agents may execute various behaviors appropriate for the system they represent-for example, producing, consuming, or selling. Repetitive competitive interactions between agents are a feature of agent-based modeling, which relies on the power of computers to explore dynamics out of the reach of pure mathematical methods. At the simplest level, an agent-based model consists of a system of agents and the relationships between them. Even a simple agent-based model can exhibit complex behavior patterns and provide valuable information about the dynamics of the real-world system that it emulates. In addition, agents may be capable of evolving, allowing unanticipated behaviors to emerge. Sophisticated ABM sometimes incorporates neural networks, evolutionary algorithms, or other learning techniques to allow realistic learning and adaptation.

There are several advantages to using agent-based models. The primary benefit is that agent-based models are able to capture emergent behavior [22]. Agent-based models are most useful in situations where agents have complex, non-linear, discontinuous, or discrete interactions; when the agents relative geographical position is of

importance; when individuals within the agent population are different and are expected to behave differently; when the interactions between agents are heterogeneous and complex; and when agents exhibit complex behavior (including but not limited to learning and adaption) [22]. Because virtually all defense missions meet all of these criteria, it is fair to assume that ABM is an appropriate technique for modeling mission performance. However, of all the modeling techniques discussed so far, ABM is the most complicated and the most time consuming to construct and run. As models become more complex, the computational resources required increase quickly [22].

Despite the advantages of ABM, it has some characteristics that make it unsuitable for executable architecting, including the disadvantages listed above. While the modeling allows for a high-fidelity model of the architecture, a significant investment in customized code would be required to fully model the alternative space. Additionally, the framework of ABM does not lend itself to autogeneration of the complete code required, although autogeneration of agent types may be possible. Additionally, ABM requires more computational resources and has a longer run time that the other modeling types discussed here, which makes it less desirable for use in early phase acquisition. ABM can, however, be applied to a handful of down selected architectures to verify the predictions of other models or to increase the certainty of predictions in order to differentiate between similarly performing alternatives.

3.5.2 Rapid Architecture Alternative Modeling (RAAM)

The Rapid Architecture Alternative Modeling (RAAM) technique is intended to be a 1st-order, parallelize-able, low fidelity analysis technique specialized for the rapid exploration of extremely large design spaces, which utilizes best-practices from computer science combined with sound principles for data specification and collection to provide a unified way of creating models for the analyst and ensures that each alternative generated is self-consistent and maintains basic mission capability. RAAM

attempts to simplify the complex generation and evaluation of SoS alternatives by creating a decision-based task hierarchy which generates operational alternatives in the same way as system alternatives, and by using a declarative notation. RAAM is parallelize-able, and can be used with cloud computing to decrease analysis times [87].

RAAM uses a simple text input file comprised of task-to-task mappings, system-to-task mappings, system-to-metric mappings, and task-to-system-to-metric mappings to specify a group of architectural alternatives. It then automatically computes feasible alternatives within the space, and evaluates them according to user-specified aggregation and transformation functions on the metrics. For the purpose of RAAM, aggregations are those functions that map many-to-one, while transformations are those functions that map one-to-one. As an example of how this works, assume that a metric of interest is the time required to complete an activity sequence. Then, time would be an element in the task-to-system-to-metric mappings, where for every system performing each task an estimate is made as to the time required for that system to perform that task. Then, as the activity thread is built through the task-to-task mappings, and as systems are assigned to each task to create a single architecture alternatives, the times for each system-task pair are summed to estimate the time for the activity sequence. A graphical interpretation of the inputs required for RAAM is shown in Figure 40 [87].

RAAM has several advantages that make it desirable for use in CBA. First, its rapid run times allow a large number of alternatives to be generated and run very quickly. In a proof of concept study, RAAM was able to create over 700 million alternatives and evaluate these against four metrics (over 3 billion model evaluations) in approximately 30 minutes using 16 compute instances [87]. Furthermore, it has been found that RAAM scales linearly (as O(n)), which is desirable since real CBA alternative spaces may be much larger [86]. RAAM also allows for any metric that

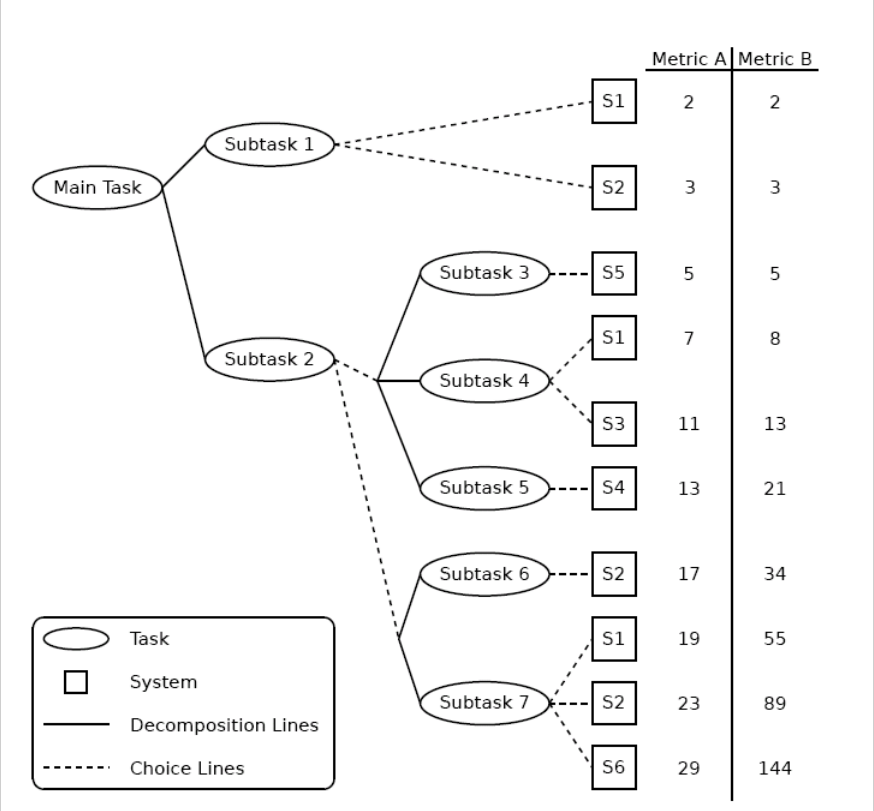


Figure 40: Graphical Depiction of RAAM's input structure [87]

can be described with aggregations and transformations to be captured, meaning that many different metrics can be estimated using RAAM. The minimalist input structure reduces the amount of information that must be gathered to use RAAM, which is advantageous both because there is little information available early in the acquisition process and because data gathering can be cumbersome and can quickly grow to be prohibitive with many modeling techniques.

3.5.3 ARCNET

The Architecture Resource-based Collaborative Network Evaluation Tool (ARCNET) is a modeling tool created by Domercant [59] to evaluate and compare different alternatives for the information networks in an SoS. It first enumerates force structure, IOL, and collaboration structure alternatives for each architecture, then models the impact of these parameters on mission success through a method based on work by

Perry in [145]. ARCNET attempts to quantify the benefits and detriments of increasing knowledge through changes to the interoperability, resource exchanges, and force structure of an architecture. ARCNET uses Perry's model to assess the benefits of collaboration on engagement outcomes. Perry would then use a measure of complexity which is based on the total number of connections between systems to capture the negative effects of collaboration, but as he himself acknowledges the inefficiencies in this approach, ARCNET instead uses an alternate complexity measure developed by Domercant. This is based on the complexity measure used in ARC-VM, which is introduced briefly in 3.6.2. Ultimately, ARCNET can be used in conjunction with an engagement model or other mission simulation to incorporate the effects of increased net-centricity on engagement outcomes. Domercant couples this with a simplified engagement model for SEAD, but any other mission performance model could be used if the appropriate mapping between collaboration and mission effectiveness is identified. For details on the mathematical implementation of ARCNET, the reader is referred to [59].

3.5.4 Existing Executable Architecting Techniques using DoDAF

There are several existing techniques for executable architecture modeling that use DoDAF, including the Executable Architecture Analysis Modeling (EAAM) capability, the Process Architecture and Analysis Model (PAAM), and the Executable Architecture Methodology for Analysis (EAMA). Dr. Alexander Levis has also performed much research on the use of DoDAF to create executable architecture models using a structured design approach, with many of his models using colored Petri Nets, although he does not use a single, named technique for doing so. Mittal has developed a framework for the semi-automated linking DoDAF to discrete event modeling. A brief summary of each of these techniques will be provided below.

EAMA was developed by Pawlowski et al. [143] at the MITRE Corporation,

based on the observation that architecture frameworks such as DoDAF provide static products that do not enable dynamic analysis of behavior and performance of a system in its operational environment over time [144]. The objectives of EAMA are to convert static architectures into executable models, develop a 'federation of simulations' for different mission threads under different operational environments using different communications networks, generalize the methodology to work with various kinds of models, and examine the resource costs of executing mission threads [143]. The methodology itself is not publicly documented; however, the application of this methodology to the dynamic analysis of C2 systems has been published. In this study, a Timed Petri Net was used to model the activity flows (or business processes). Using Popkin's System Architect as the tool for architecture development, the OV-2, OV-3, OV-4, OV-5, SV-1, and SV-2 were extracted and imported into the Integrated C4ISR Analysis and Management System (ICAMS) which was used to generate the information to populate the business process model (leveraging Petri Nets) and a network model (called NS2). Furthermore, ICAMS was used to coordinate the information in the DoDAF products with a combat simulation tool (Eagle). When the models were implemented in the three respective modeling tools, entity and relationship mapping was done across tools to ensure consistency. Measures of Effectiveness (MoEs) and Measures of Performance (MoPs) were identified either as outputs of the existing models or metrics that were a combination of the outputs of one or more models. The combat simulation was used to gather Measures of Force Effectiveness (MoFEs) [144].

While this approach has the advantage of using integrated modeling and a multi-mission approach, there are several challenges with using this approach. First, many of the tools on which the process relies are MITRE developed and access is controlled. Second, the MoPs, MoEs, and MoFEs are dependent on the modeling and simulation tools, rather than on the needs of the user. This presents a challenge in cases where

the tools are unable to provide the required information. Additionally, the creation of the executable models is not fully automated, and requires a significant programming effort to integrate the models once they are created from the architecture products [144]. Finally, while this approach was designed to be able to examine multiple alternative solutions during the execution of the JCIDS process, it was still only designed to handle a small number of alternatives, and its ability to scale to large quantities of alternatives is questionable. Since architectural changes will lead to the need to repeat the process of model creation, and thus the process of mapping the models, it is unlikely that many operational architecture changes could be explored. Thus, this methodology is not appropriate for use in the context of this research.

PAAM was developed by the Joint Forces Command (JFCOM) for experimentation [69]. The method was applied to a study in which JFCOM wished to reorganize experiment management teams to determine if changes to either the personnel or process could improve the performance of these teams. The method was designed around the use of several Gensym software packages for business process modeling. The use case of the methodology focused on reorganization of personnel. The implementation used a SQL database to provide the information necessary to input into the business process model. A custom JAVA utility allowed for this interface to be automated. Furthermore, the implementation was able to interface with Microsoft Outlook to get schedule information for team members, and output to Microsoft Project to display the suggested work flow [71]. While the framework was a fully executable model, it relied heavily on specific software and was not generalizable to other modeling tools. Furthermore, the methodology itself is not well documented in the public literature.

EAAM was being developed as part of the thesis work of Dr. Johnny Garcia [70]. According to Garcia, EAAM "will enable an organization to conduct dynamic, persistent, extensible, measurable, repeatable and interactive testing"[72]. While there is not enough published on EAAM to be able to execute the technique, the general

process is create a series of architecture descriptions, beginning with operational activity models, then mapping systems to these models, then developing the logical architecture and data flow, and using these architecture descriptions create an executable discrete event simulation based on the executable discrete event simulation technique presented by Mittal in [124]. EAAM uses the OV-1, OV-5, OV-6c, SV-1, SV-4, SV-5, SV-6, SV-7, SV-10c, and additional system requirements to create an integrated architecture from which to develop executable models [70]. However, it is unclear from the published literature how these views are integrated to create the executable framework, as well as the level of automation used in the creation of the executable models.

Dr. Alexander Levis's System Architectures Laboratory at George Mason University has done significant research in architecture modeling and executable architecting since the late 1980s. As early as 1991, Levis had demonstrated the use of Colored Petri Nets for modeling distributed intelligence systems [111]. He has since made significant advancements in the application of Colored Petri Nets to modeling architectures. He uses the AV-1, OV-1, OV-4, OV-5, OV-6a, OV-6b, OV-6c, OV-7, and sometimes the SV-7 DoDAF products to develop executable models using Colored Petri Nets. However, he states that the information contained within DoDAF is not enough. He recommends sets of initial conditions, scenarios which characterize changes in performance parameters over time, and additional rule are a minimum set of information that would be required to create executable models from DoDAF. However, he also states that if the executable model only seeks to answer sequencing questions and perform state space analysis, only a set of initial conditions and a scenario are required. [3] A full discussion of Levis' applications of Colored Petri Nets are outside the scope of this thesis, but more information can be found in [112, 180, 17, 135]. More information on his application of Colored Petri Nets to architecture modeling. However, the creation of the Colored Petri Net from the architecture diagrams in

Levis' executable model is still a manual process.

Mittal has developed a semi-automated approach to creating Discrete Event System Specification (DEVS) models from DoDAF products. In order to accomplish this, Mittal proposes two additional architecture views. First, he activity components document describes the activities as components with defined interfaces which have a logical structure. Second, the Activity Interface Specifications describes the interfaces between the activities and the entities and the execution responsibilities. Mittal uses almost all of the DoDAF v1.5 products in the development of the DEVS model, with the exception of the SV-8, SV-9, AV-1, and AV-2. UML is used to represent the DoDAF products, and then are translated to DEVS using XML [124]. This technique has many advantages. The transition between the data model and the simulation is semi-automated and allows on-the-fly changes to simulations. The techniques results in models which can be run in parallel. It can have a stochastic implementation, which allows for uncertainty to be considered. Despite the many advantages, however, the information required to use this technique is well beyond what can be expected to be available during a CBA. However, a reduced discrete event model based on a subset of the DoDAF products may be applicable to CBA, particularly in cases where mission execution time is an important metric.

3.6 Existing Techniques for Decision Support

Decision making theory (also called decision theory) is a broad field that has been studied by a wide range of disciplines. There are three groups in particular that have historically focused on decision making. First, mathematicians have explored decision making from a logical or rational point of view, examining how decisions should be made in order to comply with some set of fundamental rational behavior laws. Psychologists, on the other hand, have studied how decision-makers actually make those decisions, and what cognitive processes are used in decision making.

Furthermore, psychologists have explored whether decision-makers usually behave rationally, and if not, what can be done to reduce biases and enable more rational decision making. Finally, methodologists examine how this is all brought together to improve actual decision-making [16]. In fact, as complexity in decision-making has increased, the field of decision engineering has emerged to find, merge, and apply best practices to organizational decision-making. Since it is the application of decision-making that is of interest to CBA, it is the research of the methodologists that will be of interest here. Because decision-making is not a focus of this thesis, a full literature search is not of interest here. Rather, an overview of accepted principles to support improved decision-making under uncertainty will be presented, and will applied in the development of the CBA methodology, and sound decision-making principles will be recommended as part of the application of the methodology. Sound decision-making principles will be important because many of the solutions may meet the stated technical goals, so downselecting to a single, technical approach as is required by the defense acquisition requirements will require balancing the specific needs and requirements of the multiple stakeholders to find a solution which both meets technical requirements and satisfies all stakeholders.

3.6.1 Ullman's Criteria for Engineering Decision Support Systems

Ullman, who is a thought leader in design processes and decision theory and has published several widely referenced papers and books on the topic [165], has attempted to characterize the attributes of an ideal engineering decision support system. A brief description of each of Ullman's criteria will be included here. The first attribute, that the tool should support inconsistent information, refers to inconsistencies caused by three sources. The first source of inconsistency comes from the differing viewpoints of multiple decision-makers. A common way of handling this is to average or otherwise combine the viewpoints of decision-makers to reach a consensus. This is an

insufficient way of handling viewpoint differences as it is sure to violate Arrow's Impossibility Theorem, which is detailed in [11], and essentially states that when multiple voters are presented with three or more alternatives, there is no way to combine the votes into a single ranking which meet certain conditions of fairness. Thus, Ullman argues that engineering decision support systems should 'honor' this diversity by collecting and keeping diverse viewpoints and showing results based on each viewpoint individually rather than trying to combine them. The second type of inconsistency noted by Ullman is evaluation inconsistency. This occurs when multiple parties are asked to evaluate alternatives against the same criteria and return differing estimates of this performance. Since this variation comes from a variety of prior experiences and technical viewpoints, Ullman suggests that decision support systems should be able to handle information stochastically in order to capture the range of assessments provided. Finally, the third type of inconsistency is abstraction inconsistency, which arises due to the mixture of qualitative and quantitative information that inherently makes up the decision space in most problems. In cases where criteria are not based on physical laws, it may not even be possible to obtain quantitative estimates for these criteria. Ullman asserts that good decision support frameworks can handle both qualitative and quantitative assessments and include them in the decision process [164].

The second criterion in Ullman's list is to support incomplete information. This incomplete information may come in two forms. First, it is uncommon that the alternatives and criteria are described and mapped completely. Techniques such as QFD attempt to address this issue, but are still unable to assure that all criteria and alternatives have been included. Thus, Ullman asserts that an ideal decision support system should be able to easily add new alternatives and criteria. The second source of incomplete information is that the alternatives themselves are not necessarily evaluated against all of the criteria. This is similar to the partially described alternatives

discussed in 2.1.1. Thus, a good decision support system should be able to handle partially described alternatives [164].

The third attribute is straightforward: support uncertain decision-making information. As has been discussed in previous sections, uncertain information is a certainty, particularly in early phase SE. Therefore, the decision support system must be able to help manage and account for uncertainty [164].

Fourth, the decision support system should be able to support evolving decision-making. As more information is obtained, the criteria changes and the alternative space is constantly being refined. Decision-making is not the product of single decision event, but is rather the product of a series of decisions made over time as information evolves. Although the JCIDS process is punctuated by several decision-making events, it is overall evolutionary. Furthermore, in order to make a decision at Milestone A, a large number of alternatives must be discarded, and the process to do this will be, by necessity, evolutionary. Thus, the decision support tool should allow for information to evolve throughout the process [164].

The fifth criteria presented by Ullman is to support the building of a shared vision. This refers to the level of information sharing enabled by the decision tool. The decision tool should help to ensure that all decision-makers are working from a common set of information, rather than each having their own separate set of information from which to draw. This shared information should include the alternatives, the criteria, and the evaluations of the alternatives with respect to the criteria [164]. Sixth, Ullman states that the decision support system should provide the ranking, rating, and risk for each of the alternatives. He argues that a ranking is a simple result that helps users choose the best from a list. He further states that by including the rating, by which he means rated by the level of satisfaction ranging from 0 to 100 percent, it is possible to see if and when top alternatives are weak against the criteria and thus no alternatives are sufficient. Including the risk would allow users to see when selecting

an alternative has a high possibility of achieving the expected performance , or may become costly or untimely in its implementation [164]. This however, is in contrast with the information presented with respect to the first attribute, that the decision support system should not combine decision maker preferences and should be able to handle diverse and inconsistent opinions. In order to create a ranked list, it would be necessary to combine the opinions of decision-makers and the performance against a wide range of criteria, thus violating Arrow's Impossibility Theorem. Therefore, while Ullman may include a ranked list as one of his criteria, this work will not recognize that criteria as necessarily being desirable in a decision support system.

The seventh attribute presented by Ullman is that engineering decision support systems provide direction for additional work and what to do next. Since, as was stated earlier, consistent, complete, and fully evolved information is a rarity in decision making, it is necessary that the decision support system help decision-makers determine which alternatives to explore more fully, and what information needs to be obtained with regard to those alternatives. It is suggested by Ullman that support for what to do next come in the form of sensitivity analysis. Ullman also notes that this criteria is slightly different than the preceding six, as it refers to the decision-making process rather than the decision itself [164].

Next, Ullman suggests that decision support systems should require a low cognitive load. However, the challenge of this has be recognized. In fact, Ullman asserts that no attempts to reduce cognitive load in decision-making have been successful at doing so. He goes on to say that an ideal decision support system should help a decision-making team reach a better decision than they would have without it, but should not require an increase in the cognitive load on the decision-makers [164]. Paraphrased, while the decisions may still be difficult to make, the decision support system should not contribute to the difficultly, and if possible, should work to reduce it.

The next criteria is the support of a rational strategy. This criteria has been

developed as a result of several studies [61, 121] which have shown that early attention to developing the criteria and goals lead to a better result which is delivered in a more timely fashion [121]. also found that better evaluation of the alternatives leads prior to changing or discounting them leads to a better result. From this, Ullman concludes that an ideal decision support system should help to develop criteria and develop multiple alternatives in a flexible manner, and then enable the evaluation of alternatives against the criteria [164].

The tenth criterion presented by Ullman is that the decision support system should leave a traceable logic trail for justification and reuse. The need for traceability has already been discussed with regard to the needs of CBA. Ullman cites several reasons that traceability is important. Firstly, if the decision process is captured the information can be reused, and the decisions can be revisited and reviewed by a third party if necessary. He also cites the fallacy of relying on human memory to recall decision logic, as humans have been shown to remember things incorrectly, including what may have been said, what rationale was used, and which points were important. Thus, the decision support system should help to capture the information and logic to reduce dependency on human memory for reconstructing or justifying the decision process [164].

The final criterion presented by Ullman is that an ideal decision support system should support a distributed team. This criterion is less relevant to this work than the other criteria, as it is more of a software engineering consideration than a philosophical one. This criterion suggests that the decision support system be designed to enable collaboration between people separated by time and space [164].

As the decision support portion of this methodology is developed, it will be necessary to keep in mind nine of Ullman's criteria which apply to this work, which are:

1. Support inconsistent decision-making information

2. Support incomplete decision-making information

3. Support uncertain decision-making information

4. Support evolving decision-making information

5. Support the building of a shared vision

6. Suggest direction for additional work, what to do next

7. Require low cognitive load

8. Support a rational strategy

9. Leave a traceable logic trail

3.6.2 ARC-VM

The Architecture Real Options Complexity-Based Valuation Methodology (ARC-VM) was developed by Domercant [59] to help perform informed tradeoffs between cost, schedule, and performance during early-phase systems engineering. There are two key elements to this approach. The first is the introduction of a complexity measure specifically designed to measure the complexity in a military SoS. This is important as complexity has been shown to be correlated with cost and risk for large acquisition programs, and can act as a surrogate for these when more detailed information is not available, such as in early-phase acquisition. This is not to say that complexity is necessarily undesirable. Complex systems and SoS can often achieve higher levels of performance on difficult tasks. However, when contemplating the acquisition of a complex system or SoS, it is necessary to understand the complexity and evaluate it against the expected performance gain to determine if the increased cost and risk is worth it, and also to enable managers to take appropriate actions to plan for and mitigate these costs and risks and increase the probability of program success. It is these needs that inspire the second piece of ARC-VM, in which this

complexity measure is combined with time-valued performance estimates from early alternatives evaluations in a Real Options based visual framework to aid decision makers in acquisition decision-making.

Domercant's complexity measure that is used in ARC-VM has four fundamental elements; the functional elements, which include the Functional Domain Complexity (FDC) and the Functional Processing Complexity (FPC), and the resource elements, which include the Resource State Complexity (RSC) and the Resource Processing Complexity (RPC). The FDC describes the complexity of individual systems based on the distribution of functions across the systems. The FPC captures the complexity due to system-to-system interactions that come about as a result of performing task sequences. The RSC measures the complexity resulting from various types of resource exchanges within the architecture. Finally, the RPC captures the increases in complexity resulting from increased collaboration. These four elements are combined using equation 11 to provide an overall complexity measure for military system of systems [59].

$$C_\alpha = [FDC \times FPC]_{\mathcal{F}} \times [RSC \times RPC]_{\mathcal{R}} \tag{11}$$

The actual measurement of the FDC, FPC, RSC, and RPC has been developed by Domercant and can be found in [59].

The Real Options framework utilized by ARC-VM leverages the 'tomato garden' Real Options space developed by Luehrman in [114] for comparing the value of stock options and determining stock investment strategies. The variables in this option space were mapped to defense acquisition variables, and and combined to create an overall estimation of the acquisition value. The architecture complexity is mapped to the exercise price of the stock option. The time to expiration of a stock option is equated to the acquisition time ratio. The risk-free rate of return is equated to the probability of program success for an acquisition program. Finally, the variance

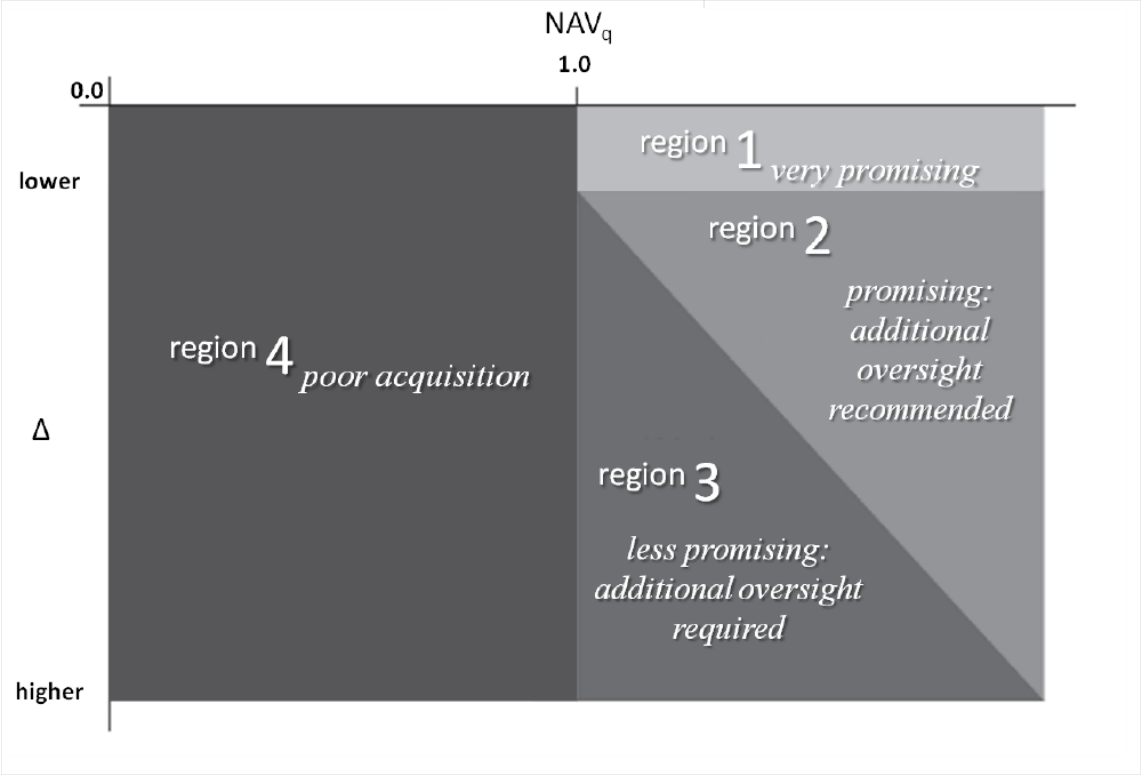


Figure 41: Domercant's Acquisition Options Space. Reproduced from [59]

of returns on stock is mapped to the variance of the overall combined MoE(s) used in assessing the SoS performance. These are combined into two key parameters, the Net Acquisition Value (NAV_q) and the cumulative deviation of effectiveness (Δ). These two parameters are then plotted in the Acquisition Options Space (similar to Luehrman's 'tomato garden'), which then gives decision makers an idea of the overall likelihood of success of the proposed program. A graphical representation of the AOS is shown in Figure 41. For details on the formalized mathematical implementation of this approach, the reader is directed to [59].

Since this approach takes into account the integration of a solution within the SoS, and provides rigorous and quantifiable way to combine performance and cost estimates, it is very applicable to helping perform decision support for a CBA, and in fact was designed with that very purpose in mind.

3.6.3 Visual Analytics

The National Visualization and Analytics Center (NVAC) defines visual analytics as "the science of analytical reasoning facilitated by interactive visual interfaces" [162]. One of the main goals of visual analytics is to battle the 'scalability challenge', which has occurred as a result of improved computer technology allowing data to be gathered and stored in previously unfathomable quantities. It has been estimated that over 23 exabytes (10^{18} bytes) of data was produced (including both streaming and stored) in 2002 alone, and that the storage of new information grows at a rate of 30 percent per year. While any one analysis problem has only a very small fraction of this amount, new modeling and simulation techniques have made it possible to have millions or even billions points of data, which can be over a gigabyte of stored information. Making sense of this information and 'finding the needle in the haystack' can be a very challenging problem indeed. Visual analytic tools and techniques are used to help people make sense of this information and uncover key insights from "massive, dynamic, ambitious, and often conflicting" datasets [162]. Although visual analytics is a fairly new discipline, with the Department of Homeland Security being credited with creating a surge of interest when they formed the NVAC in 2004, there have already been many success stories on a broad range of topics when applying visual analytic techniques. Just a few of these success stories include the development of the Scalable Reasoning System (SRS) for law enforcement, the use of Green Grid for analyzing the North American Electricity Infrastructure, and the creation of WireVis to aid finical fraud analysts [108].

One of the major goals of visual analytics is to facilitate analytical reasoning. Analytical reasoning techniques are defined by the NVAC to be "the method by which users obtain deep insights that directly support situation assessment, planning, and decision making" [162]. Ultimately, this is the ability of an analyst to uncover trends and draw conclusions from data. This is done by taking advantage of a broad

range of visual representations and interaction techniques, so that an analyst can not only see the data in multiple ways, but can also interact with the data. In fact, the NVAC states that, "Interaction techniques are required to support the dialogue between the analyst and the data" [162]. There are an increasing number of software packages being developed to meet this requirement. As full exploration of visual analytics software is outside the scope of this work, the statistical analysis package JMP® will be used here to aid in visual analytic analysis due to both its availability to the author and the author's familiarity.

CHAPTER IV

METHODOLOGY DEVELOPMENT

This chapter contains the formal development of the ARCHITECT methodology. An inductive reasoning approach will be used, in which a set of criteria is developed based on the needs identified in the preceding literature search, and each technique is then compared to these criteria. Where it is found that an existing technique exists which can sufficiently meet these criteria, the existing approach will be adopted for that step of the methodology. However, if no existing technique is found, a new approach will be developed with the goal of meeting the identified needs of the method. The reader will be taken through the logical reasoning behind each selection, and each section will conclude with a discussion of how the selected approach for that step meets the criteria identified early in the chapter. Since many of the criteria can not be formally tested, the following chapters will demonstrate the claims made in this chapter through a series of example problems. In the final chapter, the overall methodology will be assessed against the criteria based on observations made in the demonstration applications, and in cases where it is possible, analysis will be done to show that the resulting method met the criteria developed here.

4.1 Desired Characteristics of a Methodology

The primary research objective of this work, as was stated previously, is to create a capability-based systems engineering method for the early phases of design and acquisition (including gap analysis and alternative evaluation) that improves agility in defense acquisition by (1) streamlining the development of key elements of JCIDS and DODAF, (2) moving the creation of DODAF products forward in the defense acquisition process, and (3) using DODAF products for more than documentation

by integrating them into the problem definition and analysis of alternatives phases. The methodology developed as part of this research is named the Architecture-based Technology Evaluation and Capability Tradeoff (ARCHITECT) Method.

From the initial literature search presented in Chapter 1 on current acquisition requirements and criticisms of previous CBA studies, seven criteria were deemed important for the methodology to possess. However, further literature search on corporate acquisitions and existing techniques (presented in Chapter 2) led to several additional criteria being added to the list. The full list of criteria is presented in Table 6. Each step of the methodology developed in this research should be developed with these criteria in mind, and this will act as a criteria list for comparing candidate techniques at each step. This table answers Research Question 0, which asked what criteria were desirable for a CBA methodology.

A mapping of these criteria to the steps in the ARCHITECT method to which they are most applicable is shown in Figure 42.

An assessment was also done on how well the DoD performs in each area on each of these criteria according to the literature, including several GAO assessments of previous CBAs [175, 173, 174, 58, 30, 133]. The results of this assessment are shown in Figure 43. There are several things to note. First, the DoD performs very poorly overall against this list of criteria. However, as observed previously, many of these criteria are not part of the guidance for CBAs, and thus are not requirements of the current process. Furthermore, this chart represents an overall average of previous CBAs based on what is available the literature. Since most CBAs are not published, this assessment relies on the evaluations of these CBAs performed by other parties. The assessment does not indicate that all CBAs performed by the DoD are poor, just that on average, the CBAs are falling short against these criteria. Since metrics derivation is not called out as a specific step in the CBA process, it is not typically formally done in current CBAs and thus falls short against every criteria.

Table 6: Summary of Criteria for a CBA Methodology

Source Area	Criteria
DoD Acquisition	(1)The methodology should allow CBAs to be conducted more quickly (less than a year)
	(2)The methodology should result in a CBA which is transparent
	(3)The methodology should provide decision makers with an increased number and type of alternatives across the DOTMLPF spectrum
	(4)The methodology should allow materiel solutions to be evaluated with respect to multiple missions
	(5)The methodology should leverage quantitative analyses when possible
	(6)The methodology should be rigorous and repeatable, but have enough flexibility to apply across a broad spectrum of problems
	(7)The resulting CBA should include a dynamic environment that allows decision makers to interact with results (similar to an interactive design review)
WAVE Model	(8)The methodology should result in a framework to support current and future decision making that preserves the results of previous analyses and can be easily updated and allows the CBA process to be repeated very quickly once updates are implemented and a new baseline is developed
Corporate Acquisitions	(9)Integration of information and data relevant to decision makers is important, and should not be done in an ad-hoc fashion
	(10)Scenarios, assumptions, and baseline information should be consistent among all analyses conducted simultaneously
	(11)The process must include sufficient analysis and full consideration of the alternative space and be completed in a short time frame
	(12)Clarifying requirements and reducing ambiguity is important to successful acquisition
	(13)Emphasis should be placed on ease of integration of solutions into the SoS as well as performance and value
	(14)It should be verified that a new solution can be integrated in such a way that expected benefits are realized prior to choosing that solution
Cognitive Simplifications	(15)The process must help to reduce biases stemming from cognitive simplifications (the specific biases are unique to each step of the process)

Criteria	Metrics Derivation	Gap Analysis	Alternative Identification and Generation	Alternative Evaluation	Decision Support
(1) CBA conducted quickly	●	●	●	●	●
(2) Transparency	●	●	●	●	●
(3) Increased # and type of Alternatives			●		
(4) Multi-mission focus	●	●	●	●	●
(5) Quantitative analyses where possible				●	
(6) Rigorous and repeatable	●	●	●	●	●
(7) Dynamic decision environment					●
(8) Results preserved in reusable manner for subsequent waves	●	●	●	●	●
(9) Rigorous data integration					●
(10) Consistent assumptions	●	●	●	●	●
(11) Sufficient analysis of full alternative space				●	
(12) Reduced ambiguity	●	●	●	●	●
(13) Consideration for ease of integration of solutions into the SoS				●	
(14) Verify feasibility of expected outcomes			●	●	●
(15) Reduction of cognitive biases		●	●	●	●

Figure 42: Criteria Mapped to ARCHITECT Steps

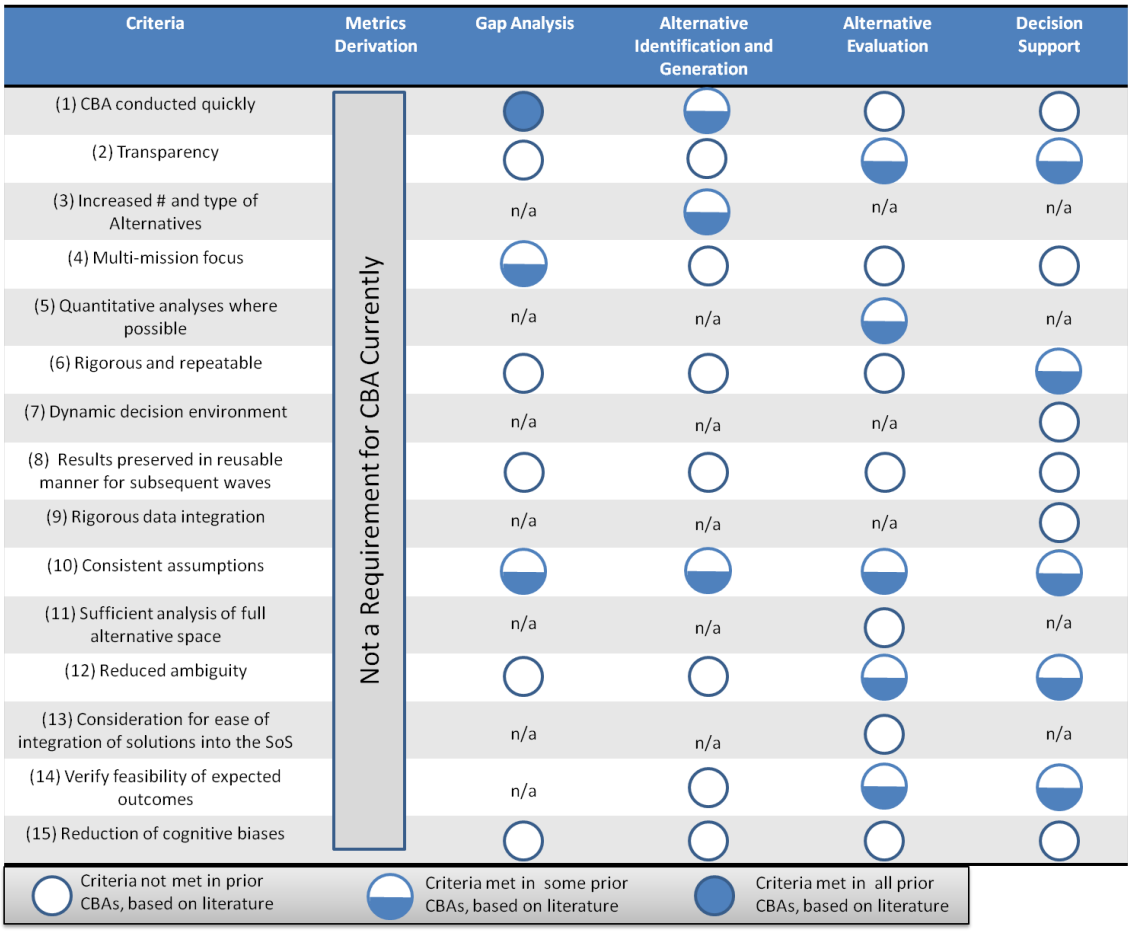


Figure 43: Assessment of Average Performance of Previous DoD CBAs Against Criteria

4.2 Formulation of Methodology

The formulation of the methodology was developed to answer the first research question, which had three parts, and is repeated below:

- (1.1) Which model of the systems engineering process, if any, is appropriate as a basis for systems engineering in the CBA decision-making process and why?
- (1.2) How should the steps of the methodology be arranged within the chosen model?
- (1.3) What are the inputs and outputs of each step, and are additional steps required to support linkages between these steps in the model?

In response to Research Question 1.1, it was determined in the literature search that the vee model was appropriate for a single incremental update to the SoS, and the Wave model is appropriate over the life cycle of the SoS. Since the methodology being developed here is for the execution of a single CBA, a vee structure will be used for the methodology. However, the vee method will be created with the Wave model in mind, such that the methodology implemented here will be able to support the wave cycle throughout the life cycle of the SoS. This means that the methodology will need to have a framework in which to preserve results and easily modify the baseline and rerun analyses for the next wave. Thus, a vee model is selected as the most appropriate structure for the ARCHITECT methodology.

Once the structure is known, the steps must be arranged in the structure, as stated by Research Question 1.2. The steps of the method are problem formulation, metrics derivation, gap analysis, alternative identification and generation, alternative analysis, and decision-support. To do this, the inputs and outputs of each step must be identified so that dependencies can be recognized and the steps can be placed in the right order. Thus, Research Questions 1.2 and 1.3 will be addressed together. Figure

Table 7: N-squared Diagram for the ARCHITECT Methodology

	(1)	(2)	(3)	(4)	(5)	(6)
(1) Problem Formulation	–	x	x	x		
(2) Metrics Derivation		–	x			
(3) Gap Analysis		x	–		x	
(4)Alternative Identification and Generation				–	x	
(5)Alternative Evaluation					–	x
(6) Decision Support	(x)					–

44 shows the inputs and outputs required by each step of the ARCHITECT process. A full discussion each step and why these are the needed inputs and outputs will be contained in the following sections in this chapter which will discuss the development of each individual step. This discussion will also elaborate on what is contained within each of the inputs and outputs. In the figure, boldface identifies those pieces of input information that do not map to an output of a previous step. Table 7 shows an N-squared diagram of how the steps of the method are dependent on each other based on the identified inputs and outputs. The N-squared diagram helps to ensure that the steps are placed in the most efficient order to avoid feedback as much as possible. In the ordering shown here, the arrangement is ideal because most of the 'x's are above the diagonal of the matrix, indicating feed forward connections. The only feedback is between gap analysis and metrics derivation, which are interdependent and therefore will require feedback regardless of how they are placed relative to each other. There is also an implied feedback between the decision support and the problem definition, owing to the fact that once a decision has been made and implemented, it will change the baseline and the problem formulation for the next wave of the SoS life cycle. It is recognized that in the process of executing the ARCHITECT methodology there will likely be unforeseen iterations. However, this approach gives a logical order of steps that will minimize iterations and ensure that needed information is gathered prior to the initial attempt to execute any one step.

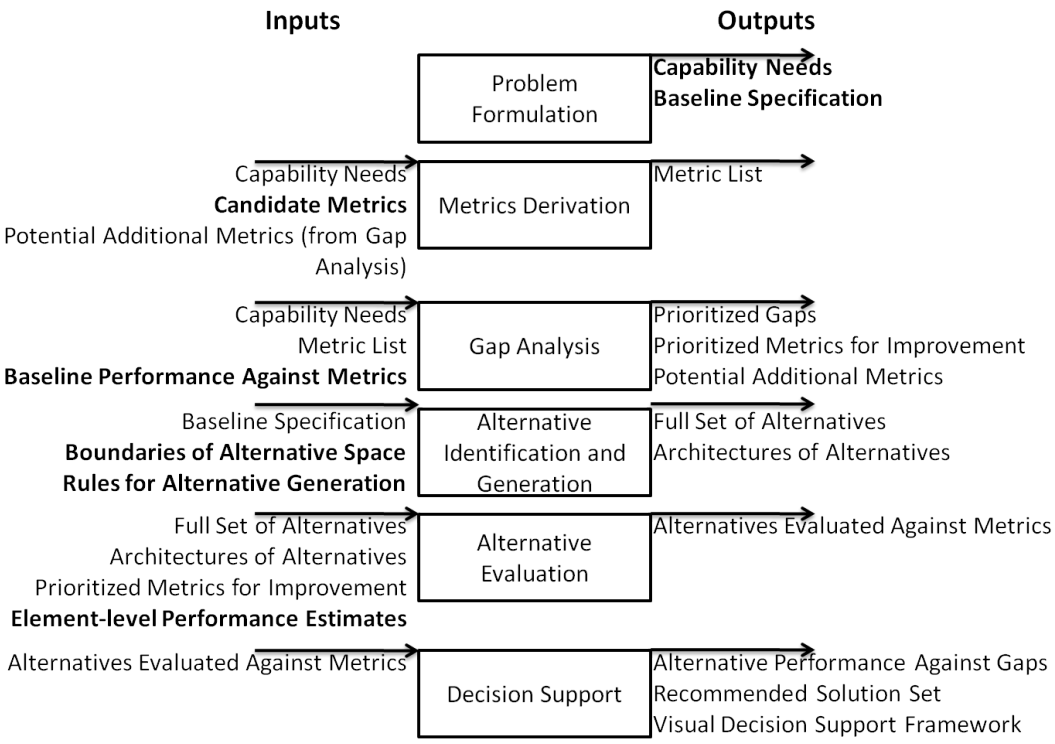


Figure 44: Inputs and Outputs for Each Step of the ARCHITECT Methodology

From the choice of a vee structure, the input and output analysis, and the N-squared digaram, the overall structure of the methodology is developed, and is presented in Figure 45. The following sections will develop each step of the methodology further and identify the sources, format, and for the required information.

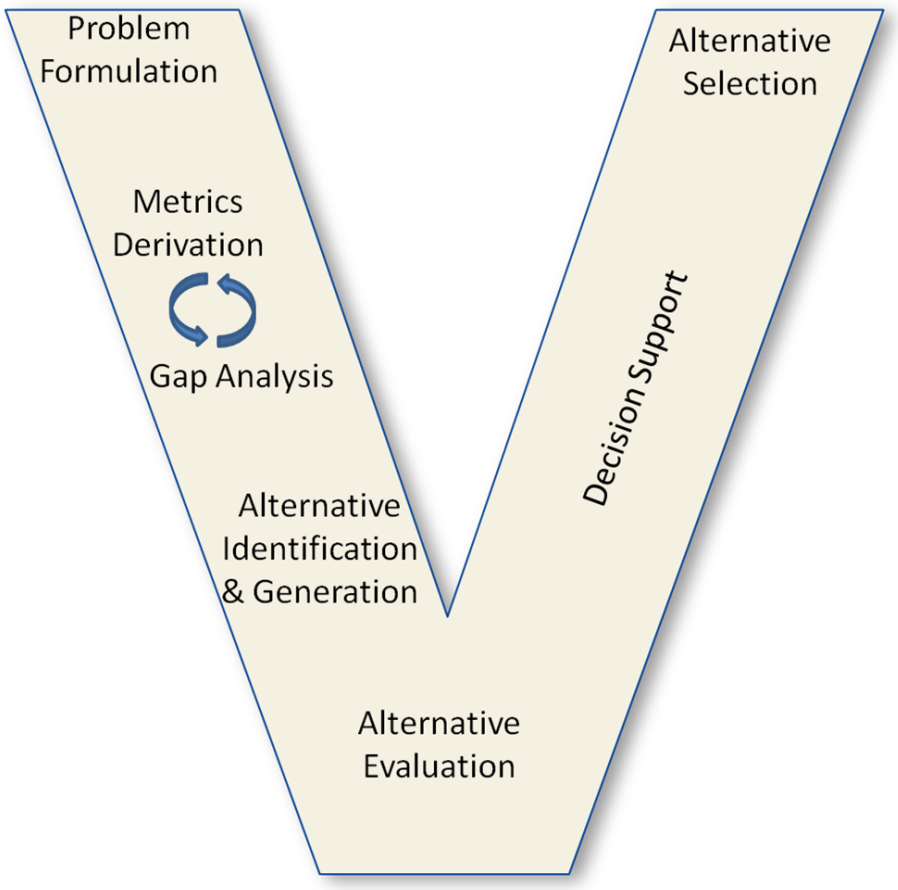


Figure 45: The ARCHITECT Vee Model

4.2.1 Metrics Derivation

The second research question addresses metric derivation and is repeated below:

- (2.1)What is a repeatable and structured way to derive a set of metrics from a top-down capability-based perspective that adequately measures baseline and alternative performance, cost, and risk?
- (2.2) How should these metrics be tested for goodness?

As was observed in the literature search, there are a limited number of well-documented techniques available to perform metrics derivation in systems engineering, and these existing techniques have much in common. In general, the techniques suggest that the process starts by identifying common information needs, which in

this case will come from the problem definition and consultation with stakeholders. Next, measures are suggested to meet those needs, and the measures are evaluated against the criteria for a good metric. The candidate measures are then collected and re-evaluated to determine if the information needs were met. If the metrics were not determined to be sufficient to meet information goals, the process is repeated. However, these techniques require a framework within which to work, to document the information needs, hierarchically decompose these needs into candidate metrics, and show in a transparent way how those metrics fulfill high-level information needs.

For this work, it is suggested that the use of the ROSETTA construct will greatly aid in the metrics derivation process. ROSETTA provides a formal way to perform the decomposition from the information needs at the capability level to the measurable values that will be used to compare alternatives. The advantage of using the ROSETTA construct is twofold. First, it provides a structure to perform the decomposition while tracking how well the candidate metrics map back to high level capability needs and including the impact of metric dependencies, which can help to reduce the total metric set and prevent redundant metrics from being tracked. By understanding how the metrics are correlated, it is possible to identify cases where improvements in one area will lead to improvements in another, and therefore track only the less expensive of the two metrics. Second, the ROSETTA framework can be updated throughout the execution of the method, and can be used to mix qualitative and quantitative data if modeling is unable to be performed for some metrics. Furthermore, the ROSETTA framework can be used to capture the baseline estimates and can also be used as the construct with which to perform the gap analysis. Because it is flexible and easy to update, it meets the requirement to be easily reused on multiple waves. It also helps to meet the criteria for transparency and the reduction of ambiguity, as the assumptions on how the metrics are used to assess capabilities and requirement fulfillment is clearly documented. Documenting the results of the

metrics derivation in the ROSETTA format will also help to create consistent assumptions in the metrics derivation as all analysts will have to come together and agree on metric definitions and mappings in order to execute the technique.

ROSETTA can easily be used to provide a multi-mission framework in which to conduct metrics derivation. In order to do this, the generic ROSETTA breakdown presented in [118] would need to be extended to the SoS level, which is easily accomplished. The SoS breakdown used in the ROSETTA framework as applied to the ARCHITECT method is shown in Figure 46. This breakdown starts and the mission level, maps missions to capabilities, capabilities to MoEs, MoEs to MoPs, allows for several levels of MoPs and necessary, and finally moves from MoPs to independent variables (IVs). Since ROSETTA provides a way to determine which parameters are most important to meeting requirements, it can also be used to help develop the list of Key Performance Parameters (KPPs) that are required at Milestone A. Use of the ROSETTA framework must be inserted into the existing metrics derivation processes, as it provides a structured and repeatable way to decompose and map candidate metrics to the high-level requirements. Metrics will still need to be assessed for goodness using the INCOSE criteria for a good metric.

In order to address the need for this step of the methodology to be rigorous and repeatable, PSM is selected to be coupled with ROSETTA to do the actual metric brainstorming and selection rather than G/Q/M. There are two primary areas where PSM is desirable over G/Q/M for this application. First, PSM suggests that candidate measures should be taken from standard sources when possible. New metrics would only be formulated when previously existing metrics are not found to assess information goals. The use of standard sources means that the process would be repeatable and the sources from which the metrics were taken could be documented to help reduce ambiguity and increase transparency. Furthermore, the DoD has documented MoEs and MoPs in the service and joint task lists [176, 103, 82, 166], providing

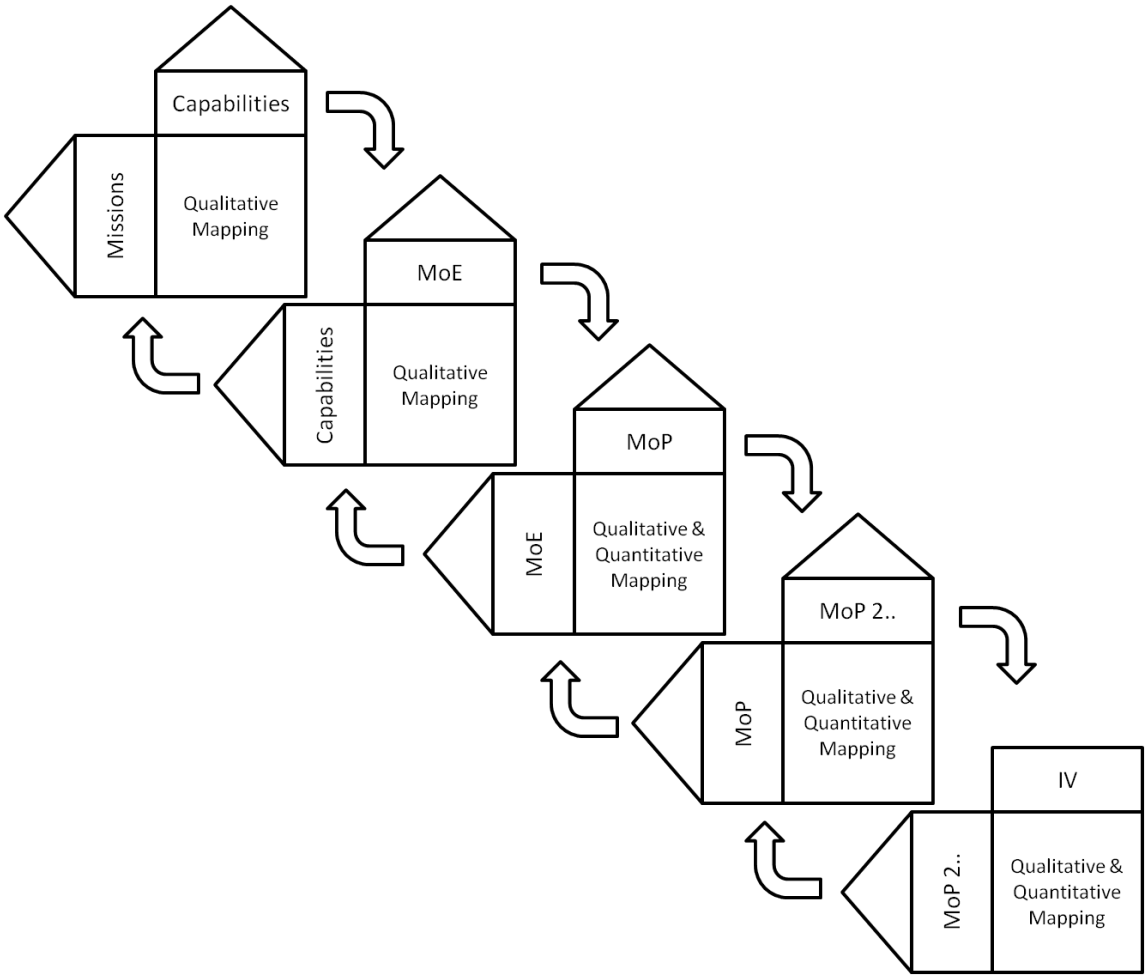


Figure 46: ROSETTA Decomposition for System of Systems

standard sources from which to select candidate metrics. The second reason for selecting PSM is that it requires that candidate metrics be assessed against criteria for a good metric, adding rigor to the process and decreasing the probability that metrics are found to be inappropriate when applied later in the process. Comparing the metrics against the criteria for a good metric will also force users to determine possible modeling strategies early and determine how the metrics would be collected. For the purpose of this work, the criteria for a good metric will be taken from [90] and [89], and include the criteria of relevant, complete, timely, simple, cost-effective, repeatable, and accurate. It should be noted, however, that an ARCHITECT user can use any preferred metrics derivation technique in place of PSM, provided that it

meets the criteria outlined in Figure 47.

For this work, the use of ROSETTA as a framework in which to apply PSM for metrics derivation is selected as the metrics derivation approach. As this is a straightforward process that could be conducted within a single workshop, it meets the criteria of conducting the CBA quickly. Furthermore, it is transparent, as the ROSETTA framework clearly shows the mappings between the metrics and the high level requirements. Multiple missions can easily be incorporated using ROSETTA, using the hierarchical breakdown shown in Figure 46. The selection of PSM contributes the rigor and repeatability, by using a pre-defined list of metrics when possible, and by ensuring that chosen metrics meet a set of criteria for a good metric in the context of the application. Because the metrics derivations and mapping are preserved in the ROSETTA framework, the results are reusable. Documenting the metrics derivation process in a single framework also helps to maintain consistent assumptions and reduce ambiguity, starting the execution of the methodology off with a common ground and a unified set of information needs. A summary of the techniques considered here and a qualitative assessment of how they meet each of the relevant criteria for a CBA based ion the above discussion is shown in Figure 47.

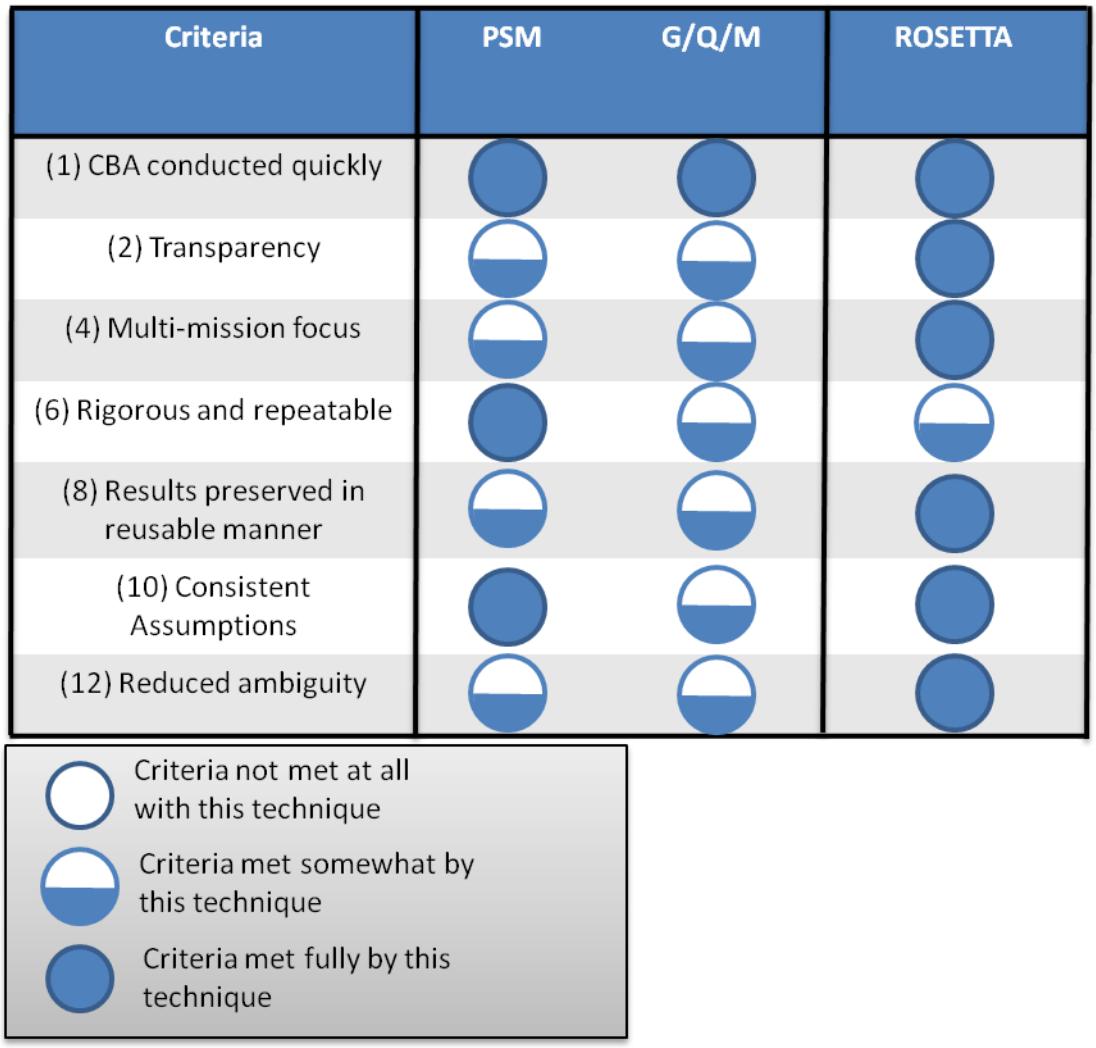


Figure 47: Qualitative Assessment of Metrics Derivation Techniques Against CBA Criteria

4.2.2 Gap Analysis

The third research question deals with gap analysis, and is repeated here for convenience.

- (3.1) What approach is best suited to robustly identify gaps in current defense System-of-Systems?
- (3.2) How should the gaps be ranked in order to capture their size, criticality to mission objectives, and uncertainty in the estimations?

Current gap analysis methods are often ad-hoc, with little emphasis on standardized processes and reusable frameworks. Although gap analysis is performed regularly, the techniques used to complete it are rarely documented in the public literature. However, the use of ad-hoc methods results in deficiencies in many of the criteria that have been developed to measure success for this methodology. Ad-hoc methods are rarely repeatable and rarely rigorous. The assumptions are unclear and undocumented, and the data is not necessarily preserved for transparency. Ad-hoc methods are highly subject to the cognitive biases of those conducting the analysis. Furthermore, the results are often ambiguous with no clear mapping to the data and information used to reach the conclusions. As such, it is desirable to find a more structured and rigorous way to perform gap analysis which will provide traceability and re-usability, will reduce ambiguity, and which will help to capture the useful experiences and knowledge of the SMEs without being subject to potential biases. For this application, the use of the ecology method discussed previously combined with ROSETTA framework can help to fill this role. Using the initial mappings of ROSETTA and adding SME estimates on how well the baseline performs in each of the metrics, the baseline performance can be projected back to the requirements space. Depending on how the initial mappings were done, it may be necessary to replace the initial mappings with sensitivity estimations. For example, for each capability, it would be necessary to give a first order estimate of how much an increase in the MoE would improve or degrade the capability. These estimations can be qualitative, but are necessary to project the baseline performance to the high-level requirements. It is likely that MoP, or even possibly MoE, values will be known from existing data for the baseline, and thus the full mappings may not be necessary. Furthermore, since the estimations of the sensitivities (and possibly the baseline performance estimates) are subject to uncertainty, probabilistic analysis will be required to project that uncertainty and obtain distributions on how well the requirements are currently being

met. Criticality of gaps will be obtained through SME estimations on the importance of missions which are then projected through the ROSETTA mappings to the capabilities to determine the criticality weightings on the capabilities.

Once this has been completed, estimations of gap size, gap criticality, and uncertainty will be obtained. The size is obtained as the percentage difference between the current performance on an MoE and the desired threshold as defined by decision makers and/or SMEs. Understanding that it is not likely to be possible to define the current performance on any MoE as a single number, it is instead defined as a range, with a minimum value, and maximum value, and a mean or mode. This mean or mode does not have to be the average of the minimum and maximum values, but should lie within the range. This value represents the SME's (or literature's) best guess at the expected performance. The criticality is then obtained by polling SMEs. However, it is most likely going to be easier for the SMEs to assess the criticality at the mission or the capability level rather than at the MoE level. Therefore, it is the necessary to project the criticality from the capability or mission level to the MoE level, which is easily accomplished using the ROSETTA framework. The criticality is estimated using an integer scale (such as 1-3, 1-5. 1-10, etc) that is chosen by the user. However, it is recommended that the integer values on the scale be given as concrete a definition as possible in order to avoid ambiguity. As examples, if assigning criticality at the mission level, the criticalities could be assigned based on how often the mission is performed, or how many other missions are dependent on it. Like the size, the criticality can be assigned a range to accommodate differing perspectives from SMEs. Like the size, it is assigned a minimum, maximum, and a mode value. However, if there is agreement, it is not necessary to provide ranges. While the approach taken here allows for the uncertainty to be accounted for as a range of values, it is not dependent on having that range. The general motivation for having the range is that the uncertainty at this stage of the decision-making process

is still extremely high, and subject to many assumptions. Thus, to ask SMEs for an estimation of a single value may be unrealistic, and is not likely to capture the full range of possibilities. Asking for a range of values allows this uncertainty to be better captured and increases confidence in the results. Furthermore, it provides an opportunity to better understand the problem space and to elicit additional information from SMEs, as a SME could be asked to justify the range by saying something like "Well, this capability has a low criticality in scenarios where..."

The size and criticality then need to be combined to generate a gap prioritization. In order to do this, size and criticality will first be normalized to a zero to one scale. Then, the largest theoretical gap would have both a size and criticality of one, and the smallest theoretical gap would have both a size and criticality of zero. The actual gaps would fall somewhere in the middle, and would have a range on the size and criticality from the uncertainty. This is shown graphically in Figure 48. Consider a to be the distance from the gap center to the largest possible gap of (1,1), and b to be the distance from the gap center to the smallest possible gap of (0,0), as shown in the figure. Assuming that size and criticality are equally weighted objectives, the largest gap would be that which has the smallest a and largest b. Therefore, the gap rating, GR, will be found by Equation 12.

$$GR = (\sqrt{2} - a) + b \tag{12}$$

The distances a and b are calculated by Equations 13 and 14, respectively.

$$a = \sqrt{(1-s)^2 + (1-c)^2} \tag{13}$$

$$b = \sqrt{s^2 + c^2} \tag{14}$$

Then, for each gap, a minimum GR and a maximum GR are calculated to determine the GR range when accounting for the uncertainty. In order to do this, a

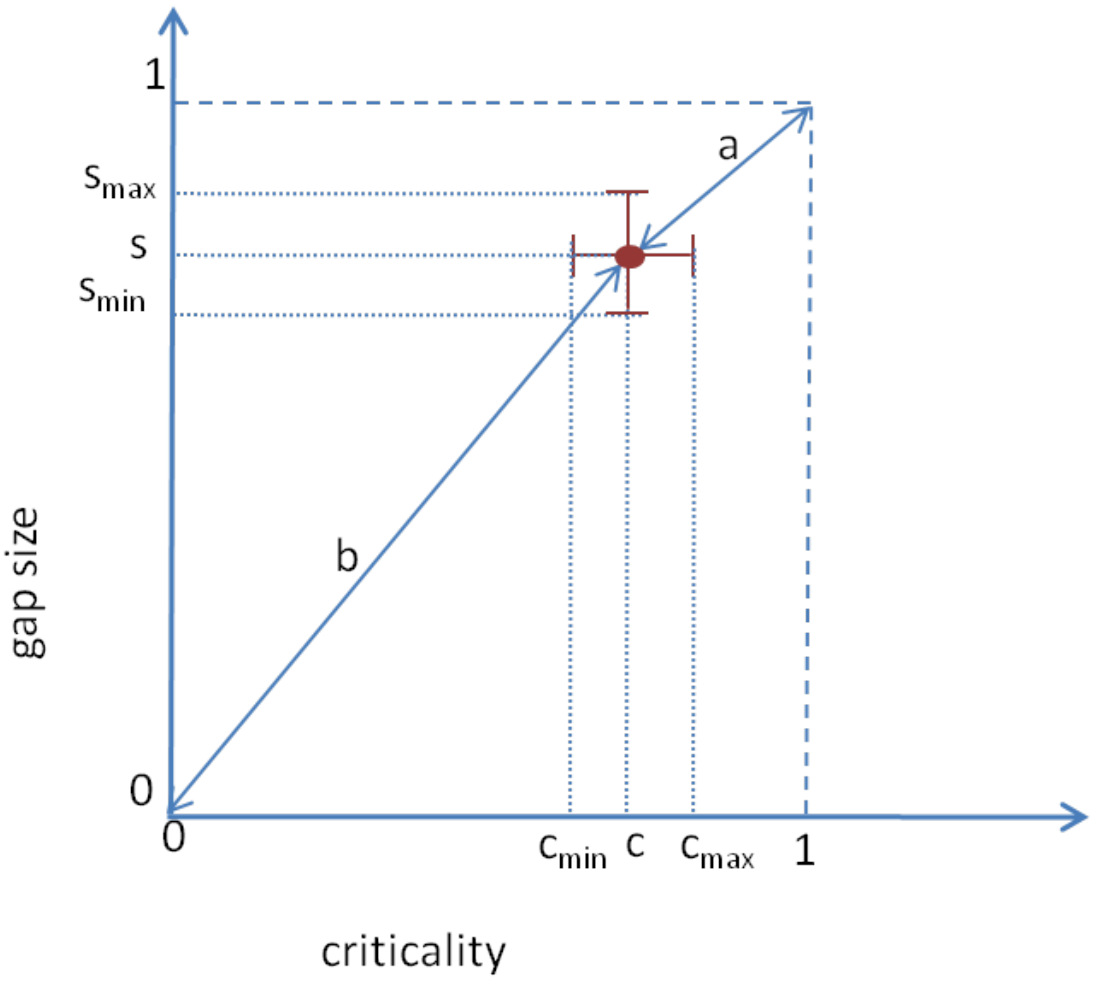


Figure 48: Graphical interpretation of Gap Quantification

maximum and minimum a and b must first be calculated. The calculations to obtain the maximum and minimum GRs are shown in Equations 15 through 20

$$a_{min} = \sqrt{(1 - s_{max})^2 + (1 - c_{max})^2} \tag{15}$$

$$a_{max} = \sqrt{(1 - s_{min})^2 + (1 - c_{min})^2} \tag{16}$$

$$b_{min} = \sqrt{s_{min}^2 + c_{min}^2} \tag{17}$$

$$b_{max} = \sqrt{s_{max}^2 + c_{max}^2} \tag{18}$$

$$GR_{min} = (1 - a_{max}) + b_{min} \tag{19}$$

$$GR_{max} = (1 - a_{min}) + b_{max} \tag{20}$$

The final result would be a chart similar to the notional one shown in Figure 49. This will allow decision makers to understand the gap rankings, and how the gap rankings may be affected by the uncertainty.

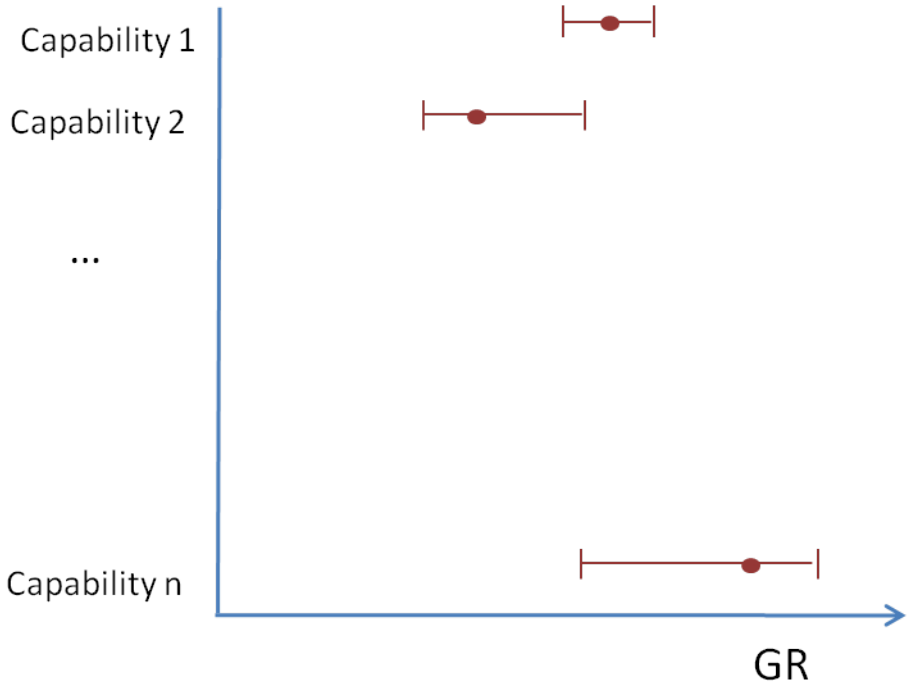


Figure 49: Notional Gap Analysis Results

The use of the ROSETTA framework with the methodology presented from ecology for gap analysis fulfills the criteria established for gap analysis in CBA. With respect to the reduction of cognitive biases, Schwenk [155] presented four cognitive simplification processes which could result in biases in gap analysis, which were discussed in 2.1.1. These include the Prior Hypothesis Bias, Adjustment and Anchoring,

Escalating Commitment, and Reasoning by Analogy. The use of ROSETTA for gap analysis helps to address all of these concerns. Because ROSETTA requires a clear mapping between variables that is reached by a consensus of SMEs (as was performed in the metrics derivation step), it is much more difficult for erroneous or poorly supported beliefs about the relationships between variables to be formed by individuals and used to make decisions. This addresses the Prior Hypothesis Bias. Because ROSETTA does not rely on SMEs or decision makers to directly estimate gap size, but rather calculates the gap size by rolling up quantitative estimates made at the metric level (from actual data when available) to determine the gap size, decision makers are less able to allow their own prior opinions on the size of the gap or other issues such as the amount of resources previously invested to influence their assessment of the gap size. This helps to address both the Adjustment and Anchoring bias and the Escalating Commitment bias. However, ROSETTA still allows the SME's expertise to be considered in the study, through the use of the qualitative mappings and the criticality estimates. However, the questions that SMEs need to answer to populate the ROSETTA framework are focused and specific, such as whether and how much a metric contributes to measuring success of a capability, or the criticality of a capability to mission success. Because ROSETTA helps to conceptualize complex problems through an organized breakdown of the problem into a sets of clear and definable relationships, it eliminates the need for applying simplifying analogies to high-level and complicated requirements to understand the problem space and identify the gaps, which mitigates the Reasoning by Analogy bias.

As was discussed in the application of ROSETTA to metrics derivation, ROSETTA is able to handle a multi-mission focus, is transparent, helps maintain consistent assumptions, and helps to reduce ambiguity. Since ROSETTA relies on actual data for which sources can be documented as much as possible, it is more rigorous and repeatable than ad-hoc gap analysis techniques. While gathering the information needed to

populate the mappings for ROSETTA for gap analysis may take longer than a purely qualitative approach where SMEs are asked to identify and rank gaps, the process should not be particularly time-consuming. Since the mappings were completed during the metrics derivation step, the estimations that are needed to to perform the gap analysis are clearly defined, making the information easier to find. Furthermore, once this gap analysis has been completed the first time, only values impacted by changes made in the previous wave will need to be reused, meaning that subsequent iterations will be able to reuse as much of the previous analysis as possible and will take less time than the initial study. A summary of the assessment of the ecology approach with ROSETTA against traditional ad-hoc approaches is provided in Figure 50.

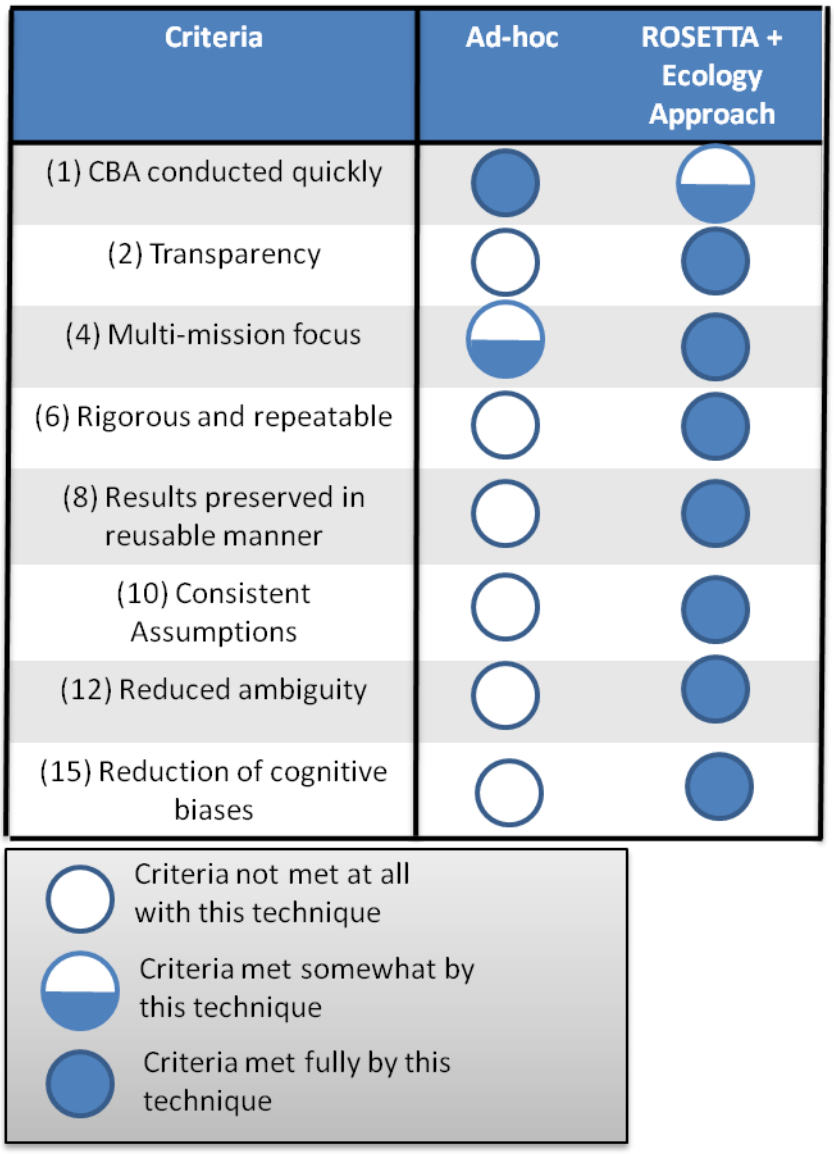


Figure 50: Qualitative Assessment of Gap Analysis Techniques Against CBA Criteria

4.2.3 Alternative Identification and Generation

Research question 4 deals with developing the alternative space for analysis, and is repeated here for convenience.

- (4.1) What is a reliable way to develop the large and diverse set of DOTMLPF alternatives for an SoS?
- (4.2) How can the alternative space be reduced to include only the subset of alternatives that are architecturally feasible?

Because it is a working assumption of this research that SoS will evolve over time, and not be designed from scratch, it is assumed that a baseline architecture is available as a starting point, and that feasible alternatives will need be able to be developed from that baseline in an evolutionary manner. This is consistent with literature on the development of SoS for the DoD [39, 57]. Thus, the alternative generation technique presented here will use the baseline as a starting point from which to make manipulations to generate alternatives. However, since it would be impossible to generate all possible alternatives manually, the technique will attempt to identify the boundaries of the alternative space, and then automate the generation of alternatives inside the boundaries that are architecturally feasible.

Furthermore, as was noted in the preceding literature search, existing approaches to alternative identification are system-focused, thus missing operational alternatives. Many current approaches are ad hoc, thereby missing large portions of the alternative space. This thesis proposes an approach to identifying both system and operational alternatives and then down-selecting a subset of alternatives to be considered in early-phase design and acquisition using the DoDAF. Using this approach, alternatives can be quickly and systematically generated and alternatives that are technically infeasible can be avoided or quickly eliminated such that they are not carried forward for analysis. This process can be used to save time and support decision making in

the early phases of acquisition. The technique consists of three basic steps. First, a baseline is identified and represented using a pre-defined subset of DoDAF products. Then, these products are manipulated according to a set of rules to create alternatives. Lastly, these alternatives are checked for architectural consistency and feasibility, and inconsistent or infeasible alternatives are eliminated, leaving only feasible alternatives for further consideration. The technique is called Technique for the Enumeration of System of Systems Alternatives (TESSA). TESSA is based off a technique previously published by the author in [75].

Defining a Baseline

Since it is the assumption that alternatives will need to be able to be achieved through modifications to the baseline, the baseline is initially developed as a starting point for TESSA. The baseline represents the current ('as-is') solution architecture and description of current operational procedures. In the rare case where no current solution exists, a baseline case is defined as any feasible architectural alternative that is able to successfully perform the mission. Beginning with a feasible solutions is critical, because alternatives will be generated by manipulating the baseline. This increases the likelihood that the generated alternatives are also feasible.

In the early phases of the acquisition process, it is unlikely that adequate information will be available to create the full set of all DoDAF products for the alternatives. Therefore, a subset of products is be selected to adequately represent the alternative space, capturing all critical aspects of the problem for generating the alternative set. The OV-1, OV-2, OV-3, OV-4, OV-5, OV-6c, and SV-1/SV-2 combined view have been selected as the necessary subset, based on the manipulations described in the following section. These views capture key information about the mission goals and operational procedures, the information needs of the SoS, the information exchange structure, and the materiel implementation of the solution. They include the

elements necessary to describe operational alternatives, materiel alternatives, organizational alternatives, and information alternatives, capturing the various elements of DOTMLPF. The first step of TESSA is to develop these products for the baseline case.

Manipulation of Products to Generate Alternatives

Alternatives will actually be generated by performing structured manipulations to these products. In order to do this, it first must be identified what manipulations are possible, and how they are dependent on each other for creating feasible alternatives. Since each product contains a specific set of information, each product has a specific set of potential manipulations. Because manipulations to any one product cannot violate the consistency of the entire architecture, it is often the case that multiple products will need to be manipulated simultaneously in order to create one feasible alternative. It is necessary to have a way of uncovering inconsistencies, so that if a manipulation or set of manipulations leads to an inconsistency between products, this can be flagged, and the alternative can either be discarded or the inconsistency can be fixed. Applying relational transforms between products that formalize the relationships between products can be used to verify the consistency of alternatives generated using these manipulations. Manipulations to DoDAF products must therefore meet two criteria. First, they must eliminate the ability of the architecture to complete mission goals, and second, they may not results in inconsistencies between products.

In order to systematically create alternatives, potential manipulations are identified and attempted one product at a time, beginning with the highest level product and moving down to the most detailed product being used. Since the OV-1 is a high level depiction of the architecture goals, it is excluded from manipulations, assuming that the highest level mission needs will remain constant. It is, however, an important product to create and capture for communication of the overall goals, and should be

kept updated as necessary as manipulations are made [46, 51]. Since the OV-1 is excluded from manipulations, the first product for potential manipulation is the OV-2. This is a high level product that states the information and resource flow needs, and many other products are elaborations of how these exchanges will occur. The OV-2 describes the operational nodes, the needlines between those nodes, and the information and resources exchanged across the needlines [46, 51]. Each of these types of information can be used as a platform for potential manipulations. These include adding or removing nodes, combining nodes, and altering information needs so that either the needlines are re-routed or the information exchanged across them changes. This, at first glance, would appear to generate a large number of alternatives. Even if the OV-2 baseline has only 3 nodes and 3 needlines, there are 6 ways nodes can be removed, 4 ways nodes can be combined, and an undefined number of ways nodes can be added or information exchanges can be changed. However, not all of these manipulations will create a consistent architecture which is able to perform the operational goals of the architecture, and therefore the number of feasible manipulations is often quite small.

After a manipulation is made to the OV-2, all of the other products must be checked in turn to determine if they must also be manipulated in some way to maintain consistency and generate a feasible solution. It is very possible that some products will have multiple potential feasible manipulations stemming from the each candidate OV-2, and each of these separate manipulations will represent a new alternative. The process described above will need to be repeated for each of the products being considered to capture the full alternative space.

Since the OV-3 is a supporting product to the OV-2, it it unlikely that manipulations to this product will generate unique alternatives. The OV-3 provides additional details on the information and resource exchanges across the needlines [46, 51]. It is necessary to include this product and update it with OV-2 manipulations, but changes

to the details contained within the product are not at the level of new alternatives in early phase acquisition. In fact, in the early phases, it is likely that the information will not even be available to fully populate the product with all details. Therefore, independent manipulations to the OV-3 will not be considered here, but the impact of manipulations of the OV-2 to the derived OV-3 will be included.

The OV-4 provides the organizational relationships, and can be used to describe who is responsible for which activity or node. Varying who is responsible for a task or node may change what activities are used accomplish the capability, but will also affect which candidate systems are available to perform that task or to reside in that node. In this way, the organizational options willing to be considered may limit available system choices later in the process. Thus, the organizations will act more as a filter on possible system-activity pairings than to generate more alternatives. The system space can be reduced or expanded depending on how a decision makers wishes to consider re-organization possibilities.

The OV-5 describes the operational activities that need to be performed to accomplish capabilities [46, 51]. This product includes the activities, the operational node in which they occur, and, in the OV-5b, the order in which they are performed. The TESSA method will use a very specific interpretation of the OV-5b. The OV-5b will contain the activities, the operational node in which they occur, and the dependencies between the activities (i.e., which activities are required to precede each other in order to maintain the capability). Changes to OV-5 will then include the addition or removal of non-critical activities, sequencing changes to the order in which node activities are performed, and possibly changes to the dependencies between activities.

The OV-5 is closely coupled to the OV-6c. The OV-6c provides the specific time sequencing of the activities in the OV-5b [46, 51]. If the level of detail available to create this product is available, it is possible to generate alternatives which include the time re-sequencing of activities. The OV-5b will provide the constraint on the ways

in which the activities can be re-sequenced. Activities can be resequenced provided that the shuffling of activities does not cause the ability to perform the mission to be violated. The OV-5b should be used to check that the new ordering does not violate the dependencies and thus the capability is preserved. If the level of detail required for an OV-6c is not available, a DSM can be used to accomplish the re-sequencing.

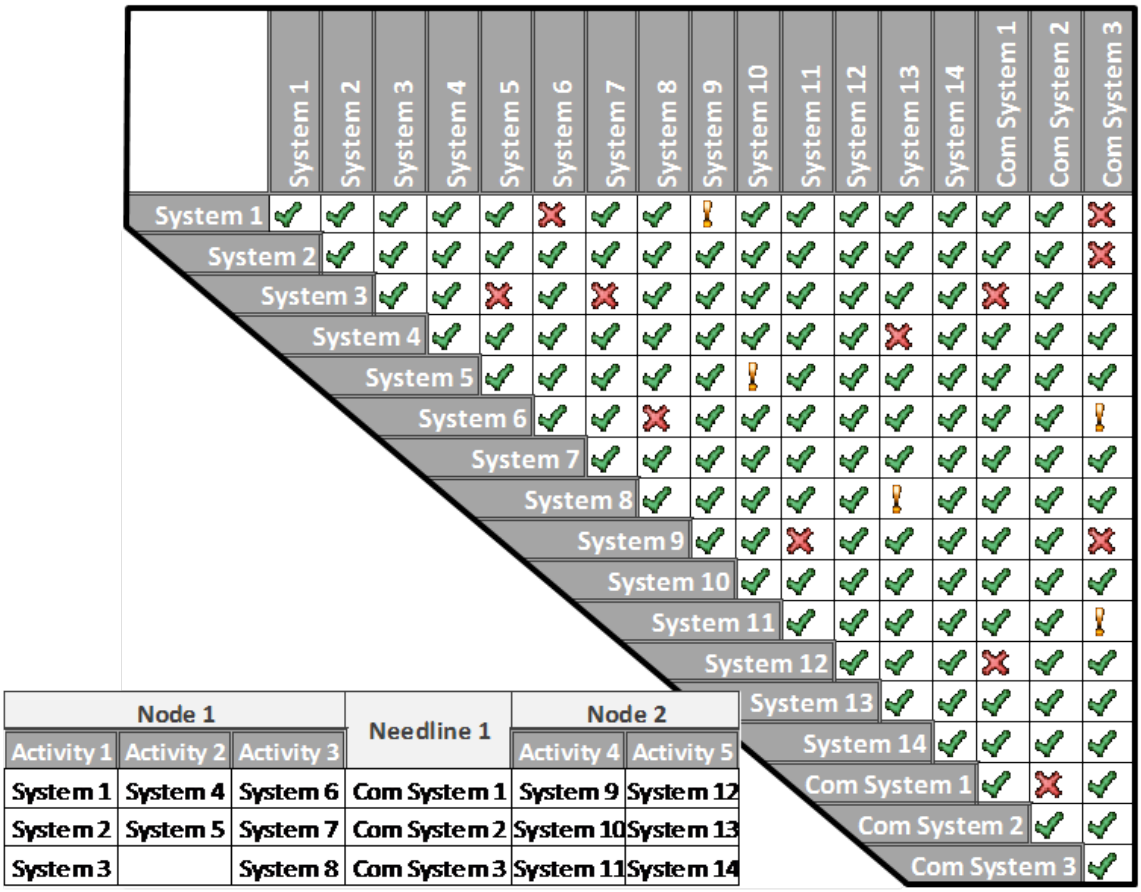
In order to generate the materiel alternatives and some of the information alternatives, a combined SV-1/SV-2 will be used. These views include the systems resident in each of the operational nodes to perform the operational activities of that node and the communication systems that are present across each needline to enable the physical exchange of information and resources. [46, 51]. These views are used to examine alternative materiel system solutions, including communication systems. Alternatives generated in this view must preserve the relationships shown in the OV-2, and enable performance of the activity sequences defined in the OV-5b and the OV-6c. Thus, the alternatives that can be generated by this product are heavily dependent on the previously created views.

A MoA and morphological analysis (discussed in 3.4.1) is used to aid in generating the SV-1/SV-2 alternatives. To do this, a matrix is created which lists each of the activities and needlines. For each activity and needline, candidate systems for performing this activity or exchanging information or resources across that needline are listed. Incompatibilities between systems that will restrict the pairings of systems in the alternative generation are identified using a compatibility matrix. It is possible to also attach additional metadata to the systems for use in filtering alternatives which do not meet decision-maker criteria. Examples of this information include the Technology Readiness Level (TRL), the organization owning the system, the cost of the system, or the highest interoperability level (IOL) enabled by that system. Using the MoA and the compatibility matrix, potential alternatives can be quickly generated by generating threads of compatible systems through the matrix. This process has been

automated by RAAM. When filters are available based on the inclusion of metadata, systems that do not meet the filtering criteria can be eliminated from consideration.

To better demonstrate the general process of generating system alternatives, a notional MoA and compatibility matrix are shown in Figure 51. The possible alternative systems for each activity and needline are laid out in the MoA, where it can be seen that each activity within each node and each needline between nodes have one or more potential system alternatives. These alternatives should include, at a minimum, the systems used in the baseline. All system alternatives are mapped against each other in the compatibility matrix, where a check mark implies that the systems are compatible and can be used together. An exclamation point implies that they can be used together with some modification or interface management. An 'x' implies that the two systems cannot ever be used together, and the choice of one prohibits the choice of the other. It is expected that this would rarely be the case in a SoS where systems are distributed and interfaces are often enabled by software. However, there are cases where systems could be incompatible, such as using a non-carrier-based aircraft with an aircraft carrier. An example thread through the MoA is shown in Figure 52. Systems that are highlighted and bolded have been selected for inclusion in this particular alternative architecture. Systems that are crossed out have been ruled out due to an incompatibility ('x') with one or more of the selected systems. Systems in italics are designated 'to be selected with caution', because they will be difficult to integrate with one or more of the selected systems. This was denoted by the exclamation point in the compatibility matrix. Systems with no designation are simply those that were not selected by the algorithm. Choosing an alternate system in any activity or needline will change the available choices in all other activities and needlines. As an example, choosing System 3 instead of System 2 would eliminate Systems 5 and 7 from consideration. All possible system combinations for the SV-1/SV-2 should be able to be created using the MoA and compatibility matrix. By

mapping the systems to the operational activities they support, a SV-5b has been created as a by-product of this process. The algorithm in RAAM acts to generate all of these threads automatically.



Node 1			Needline 1	Node 2	
Activity 1	Activity 2	Activity 3		Activity 4	Activity 5
System 1	System 4	System 6	Com System 1	System 9	System 12
System 2	System 5	System 7	Com System 2	System 10	System 13
System 3		System 8	Com System 3	System 11	System 14

Figure 51: Notional MoA and Compatibility Matrix. Reproduced from [75]

Node 1			Needline 1	Node 2	
Activity 1	Activity 2	Activity 3		Activity 4	Activity 5
System 1	System 4	~~System 6~~	~~Com System 1~~	System 9	~~System 12~~
System 2	~~System 5~~	System 7	Com System 2	System 10	~~System 13~~
System 3		System 8	~~Com System 3~~	System 11	System 14

Figure 52: Sample Threads through the MoA. Reproduced from [75]

Figure 53 shows a tree representation of the process and order of product manipulations to create the alternative space. In this notional tree, 16 alternatives are depicted. One alternative is highlighted to show that an alternative is comprised of a

full set of the products, rather than any one product in isolation. If desired, a tree like this one can help to keep track of the alternatives, and could be used to record data and information gathered about each alternative. Files could also be linked with each box in the tree, creating a framework to easily store and access different products. However, use of a tree like this will become increasingly difficult as the alternative space grows. The OV-4 is not included in the tree because it is used to filter out alternatives where the organization responsible for the task does not own a given system, and therefore is actually used to reduce the alternative space rather than to expand it.

Consistency and Feasibility Checking

As was eluded to previously, generated alternatives must be checked for feasibility and consistency. This can be done by verifying the preservation of the relationships between products. Treating the views as formal models can enable the use of transforms in the consistency checking. In the OV-2, nodes are elements of the model that are related by needlines [74]. More formally, the nodes can be treated as set, and the needlines can be treated as relationships on that set. Additionally, subsets of the total activity set occur in specific nodes. The OV-3 will provide the details of the information exchange needs across each needline. All OV-2 needlines must have at least one corresponding information or resource exchange element in the OV-3.

The set of activities in the OV-5 must support the information flow specified in the OV-2, or the consistency of the architecture is not maintained [74]. This can be checked by mapping the OV-5 activities to the node or nodes in which they occur. This mapping is already recommended in the DoDAF specification [46, 51]. The inputs and outputs of the activities (or the string of activities) should match the inputs and outputs along the needlines of that node. Otherwise, the architecture will not be able to perform its required capabilities. The OV-5 also models the

dependencies between activities, as discussed in the previous section. To formalize this as a model, consider the activities to be the elements and the dependencies to be the relations between them.

The OV-6c shows the sequencing of the activities, and thus must include all of the activities in the OV-5 without violating the dependencies specified. Thus, if activity A is required to precede activity B in the OV-5, activity A must precede activity B in the OV-6c. Otherwise, the architectural alternative should be deemed infeasible. This information can also be captured using a design structure matrix (DSM), as suggested above. While different types of DSM exist and have been suggested for use in various applications [26], a task-based DSM that captures relationships between activities would be appropriate here. In fact, the use of DSM to explore activity sequencing has been commonly used, especially in the realm of product development processes. In fact, Yassine [183] has suggested the use of a task-based DSM to help identify tasks that must be performed in series, those that can be performed in parallel, and those that are coupled. the method can handle complex task relationships, including interdependency in addition to the sequential and parallel processes that are addressed by most process management tools (such as the Program Evaluation and Review Technique (PERT), Gantt, and the Critical Path Method (CPM)) [183]. While Yassine's method was designed for process management, the application of the technique in this context would allow DSM to be used to uncover the full set of sequencing alternatives, and help these alternatives be visualized by decision-makers. Regardless of the technique used to describe the alternative space and find potential alternatives, this would need to be coupled with an automated algorithm to identify all possible alternatives and couple these with the other dimensions of the alternative space.

Lastly, the combined SV-1/SV-2 must preserve the relationships enumerated in the OV-2, and must be able to perform all of the activities in the OV-5. This means

that the systems selected must be able to perform the activities, in the correct order, and must be able to interface with each other in such a way as to enable all of the required information and resource exchanges. A model transform can be used to verify that all operational needlines are adequately addressed in the physical instantiation of the architecture [74]. Additionally, the specific information elements called out in the OV-3 must be able to be carried across the specific communication systems called out in the communication relay in the SV-1/SV-2 corresponding to the appropriate needline in the OV-3. The SV-5b generated as a by-product of the process can be used to make sure that systems are chosen to perform all activities. To do this, all that is required is to verify that each column of the MoA corresponding to an activity occurring in the alternative has a system selected. Then the SV-1/SV-2 can be used to examine which systems must interface and check that these systems are not incompatible. The described relationships are summarized in Figure 54.

Automating the Alternative Generating Process

Since the alternative space is extremely large, it not desirable to perform all of these manipulations by hand, as is suggested by the TESSA approach. Instead, it would be preferable to set up the boundaries of the alternative space and automate the actual generation of the alternatives. Using the relationship preservation rules described above will allow architecturally infeasible alternatives to be weeded out, leaving only architecturally feasible alternatives.

The RAAM framework is able to perform the automatic generation of alternatives and preserve the relationships to create only feasible alternatives, but does not include a technique for decision makers to specify the limits of the alternative space. Instead, the input is in the form of an text input file that can be generated from user specification of the alternative space. The inputs to RAAM include task-to-task

mappings that specify the operational activity sequence(s) possible in the architecture, system-to-task mappings that specify what system alternatives are available to perform each task, system-to-metric mappings that are user defined estimations of system performance that are independent of the task the system is performing (such as acquisition cost or TRL), and system-to-task-to-metric mappings that are user defined estimations of system performance that are not independent of the task the system is performing (such as time to perform task). However, a graphical interface is needed that allows users to specify the needed information, and which then translates that information into the RAAM input format. TESSA can provide this graphical interface.

However, there may be other constraints besides those included in RAAM. For example, although an F-22 may be an excellent option for engaging a heavily guarded target, if the Navy is responsible for engaging that target the F-22 would not be an option since the Navy does not own F-22s. However, it may be an option to hand over organizational responsibility to the Air Force, thus making the F-22 an option. Decision makers will need to specify any organizational constraints so that alternatives that do not meet these requirements can be eliminated. Once the matrix of alternatives has been created in TESSA, any system-task pairings that do not meet organizational criteria can easily be filtered out prior to translating the alternative space into RAAM.

Another consideration which is not considered in RAAM is the ability of any two systems to interface, as well as the level of interoperability across that interface. Interfaces are notoriously difficult, because the interfaces are typically part of the network architecture, which is often handled separately. However, it is necessary to ensure that expensive upgrades to communication systems will not result from the choice of a system, or if they do, that the cost is worth the benefits of that system. One of the challenges with handling interfaces is there are multiple dimensions to the interface

problem. First, certain interfaces are required in order to complete a mission. Using the preservations of relationships as described above can help ensure that the minimum interfaces are enabled. However, the use of other interfaces may improve mission performance, and this has been the rationale behind the push for 'net-centric' warfare. Furthermore, there are many levels of ability to interface, which is ofter referred to as the IOL of the interface. This adds an extra dimension of difficulty to assessing the interfaces, and to analyzing the performance of the SoS. This extra alternative dimension is not currently included in RAAM. The result of this is that for every alternative generated by RAAM, there are multiple additional alternatives created by varying the network architecture. However, it is likely that a limited amount of information about the interfaces will be available during CBA. Interface aspects such as the physical interface specifications and volume of information exchanged are well outside of the scope of consideration for a CBA study. However, it is possible to explore the impact of including or excluding optional interfaces and the impact of increasing the IOL across the interfaces at a first order level using network models. Therefore, these aspects of interface alternatives will included for consideration in the ARCHITECT methodology.

Since these dimensions are not included in the RAAM framework, they will need to be added to the alternatives generated by RAAM. Thus, each alternative generated by RAAM is used to spawn an additional set of alternatives that are created by varying the network structure and the IOL across the interfaces. This will be implemented based on the system portfolio of the alternative, using concepts derived from the system-to-system interoperability matrix in LISI [28]. A notional System-to-System Interoperability Matrix is shown in Figure 55.

This idea has been implemented in ARCNET, where this matrix will be broken up into two matrices, one which is used to determine whether two system types must be able to talk (based on the SV-1/SV-2 and the OV-5), and one which specifies the

IOL across the interfaces. The first matrix, which will be referred to as the interface matrix, will be populated with ones and zeros, where one represents system types that must interface, and zero represents system types that do not necessarily need to be able to interface. Alternatives with varying levels of net-centricity are then created by changing one or more of the zeros to ones in a methodical manner. In fact, by treating each zero as a variable with two settings (zero or one), an algorithm to calculate a full factorial design of experiments (DoE) can be used to generate the full alternative set. Then, for each system type, the maximum IOL enabled by the hardware and software resident in that system is recorded as part of the metrics collection for that system. Then, for every system type pairing, the minimum of the two IOLs will be taken as the default IOL for that system pair. Interoperability alternatives can be created by then increasing each of the IOLs in a methodical manner. Again, by treating each value as a variable with discrete settings between the default IOL and four (which is the maximum IOL in LISI), a full factorial algorithm can again be used to generate the full set of IOL alternatives. In addition, RAAM does not examine alternatives that are created by varying the composition of the force, also called the force structure. However, if users provide a range of quantity to be considered for each system type, ARCNET can also be used to generate the full factorial space of possible force structures. Domercant has developed a method to execute these ideas in his ARCNET tool, which is described in detail in [59].

Summary of Dimensions of SoS Alternatives and How They are Considered in ARCHITECT

For an SoS, the alternative space has been decomposed into a series of dimensions across which the architecture can be manipulated to create new alternatives. These are:

- Operational manipulations,including tasks and process flow

- System manipulations
- Interface manipulations, including the collaboration structure and the IOL
- Force structure manipulations on the quantity of systems used in mission execution
- Organizational manipulations

By combining TESSA, RAAM, and ARCNET, it is possible to fully describe the full set of architectural alternatives across all of these dimensions. TESSA provides a way to generate possible tasks, process flows, candidate systems, interface requirements, and consider organizational constraints. TESSA uses the operational constraints to filter out system-task pairings that are not feasible with organizational constraints. RAAM provides the ability to combine the systems and tasks to create the full set of system portfolios and operational implementations of each of those system portfolios. ARCNET then, for each of the alternatives generated by RAAM, adds the full set of interface alternatives (including variations in both the net-centricity and the IOL), and the full set of force structure alternatives. This is summarized graphically in Figure 56. It is important to realize that Figure 56 oversimplifies the alternative space somewhat, in that there are often multiple ways in which the same system portfolio can be used to accomplish an operational alternative. This occurs because systems within an SoS are often multi-role, and can be used for different tasks. For example, if there are three tasks, A, B, and C, and there are two systems 1 and 2, and system 1 can do tasks A and B, and system 2 can do tasks B and C, there are actually two alternatives associated with that operational alternative and system portfolio pairing. In one alternative, system 1 does tasks A and B while system 2 does task C, and in the other, system 1 does only task A while tasks B and C are done by system 2. Thus, the alternative space is even larger than is indicated by this simplified graphic.

TESSA, ARCNET, and RAAM Evaluated Against CBA Criteria

The use of TESSA, ARCNET, and RAAM, when evaluated against the criteria for a CBA methodology, is an appropriate selection for the alternative identification section. Most obviously, perhaps, the need to examine an increased number and type of alternatives as compared to past CBAs is easily met, as the techniques selected here enable millions or billions of alternatives to be considered as compared to traditional CBA which look at only a handful. Furthermore, TESSA, RAAM, and ARCNET all ensure that both materiel and non-materiel alternatives will be considered. The need for transparency is met by fully retaining the choices made by users regarding what to include in the alternative space in a single computer-readable file. This file is then run through RAAM, which calculates the full combinatorial set of alternatives based on this file. These alternatives are then fed into ARCNET to complete the full description of the alternative space. This helps to meet the criteria for rigorous and repeatable. This also meets the criteria that results must be documented in a way that is reusable for subsequent waves. Since the alternatives are all generated from a single information set that includes all of the assumptions, the criteria for consistent assumptions is also met. A standard front-end to the process provided by the use of standard DoDAF products that are then converted to a standardized format also helps to reduce ambiguity. Since RAAM is able to handle multiple missions (by treating each mission as a separate task at the highest level of the hierarchy), and the TESSA process could be performed for multiple missions in order to generate the alternative space across multiple missions, this approach is consistent with a multi-mission focus. Since TESSA, RAAM, and ARCNET all have built-in measures to ensure that the resulting architectures are technically feasible and maintain basic mission capability, the requirement that feasibility of expected outcomes is verified (to

the extent possible prior to the stage of alternative generation) is met as well. With respect to conducting CBAs more quickly, there are several advantages to the TESSA, RAAM, and ARCNET approach. First, because users do not have to fully define each alternative, but rather only the characteristics of the types of alternatives to be considered, it is expected that brainstorming sessions will be able to be conducted in a relatively short amount of time as compared to more traditional AoA brainstorming sessions such as those described in [167] where users must fully describe each concept manually. Since the description of the alternatives and actual generation is done automatically using RAAM and ARCNET, the actual alternatives are created and described very quickly (in less than a day with only a single processor based on prototype studies [87]).

With respect to the reduction of cognitive biases, there were four cognitive simplification processes which were identified by [155] to potentially cause biases. Single Outcome Calculation was the tendency of individuals to focus on only one alternative. Since the approach proposed here is designed to focus on each aspect of the alternative space in turn, with specific alternatives then being found using an automated algorithm, users never self-define a full alternative architecture. Rather, users define features of alternatives that they are willing to consider, and the alternative space is populated from this. Furthermore, because the alternatives are generated combinatorially and taken through the first stage of alternative evaluation (which will be discussed in more detail in the following section), some quantitative model-based evaluation is done prior to users having the ability to see or reject the alternatives. This helps to alleviate biases due to Inferences of Impossibility. The third simplification process, Denying Value Tradeoffs has a similar cognitive root to the previous two, in that it happens when a user has a single alternative in mind. In this case, the user will incorrectly interpret the favored alternative to have large benefits with little cost. This is combated in two ways: first, by having a process that explores large

portions of the alternative space and second (as will be discussed in the following section), by having a process that evaluates alternatives automatically based only on only the performance and cost estimations for those alternatives and forces the user to perform an initial downselection without knowing which alternatives are being eliminated. Finally, Problem Set, which reduces the alternatives considered through use of a single problem solving strategy or over-constraining assumptions, is addressed by asking the user to come up with various ways of solving the problem (operationally, with materiel solutions, organizationally, etc) in turn. This helps users to think of types alternatives that may not have been considered previously. A summary of the TESSA, ARCNET, and RAAM approach as compared to traditional ad hoc methods for alternative generation is shown in Figure 57.

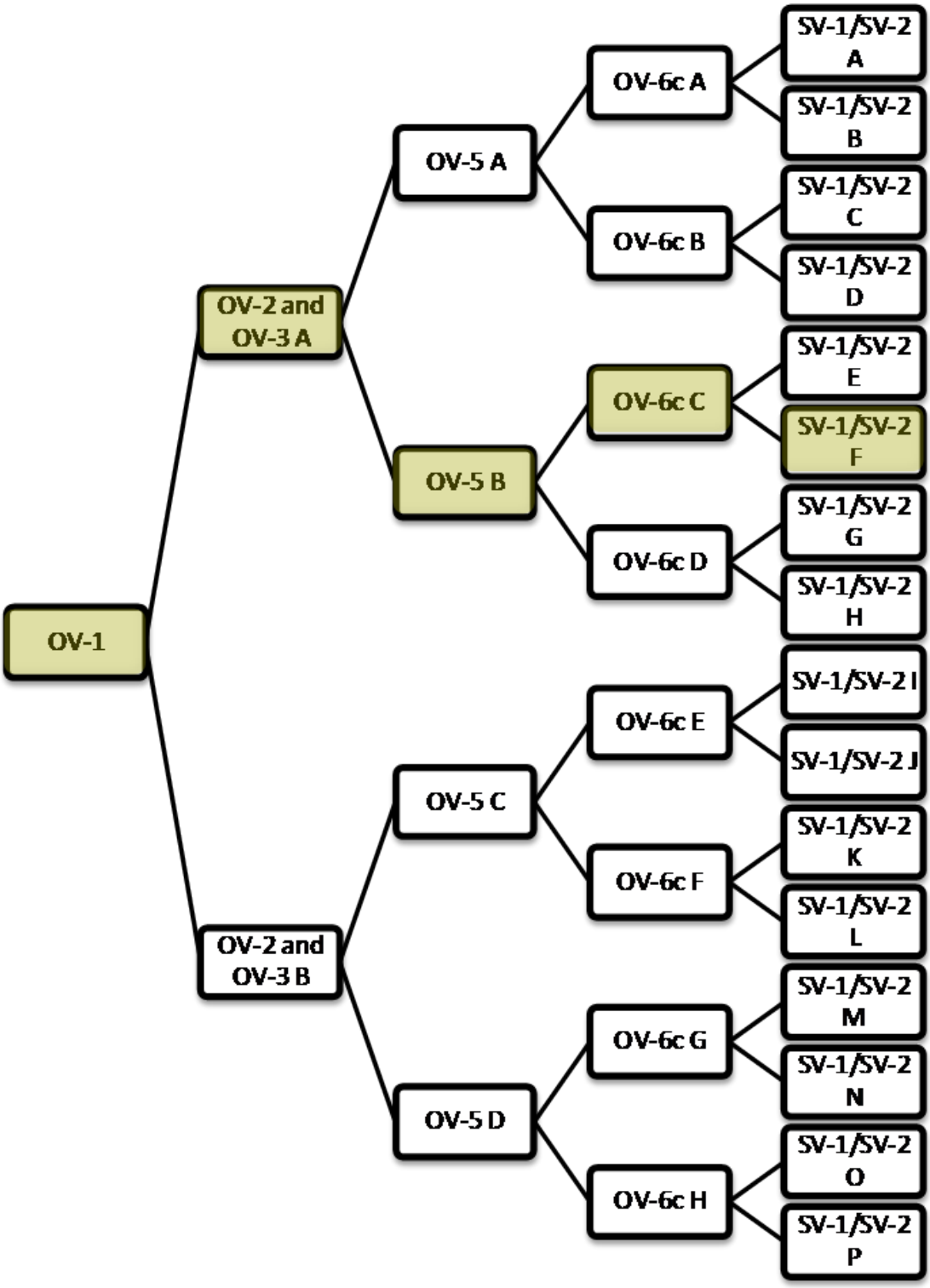


Figure 53: Notional alternative tree with one alternative highlighted. Reproduced from [75]

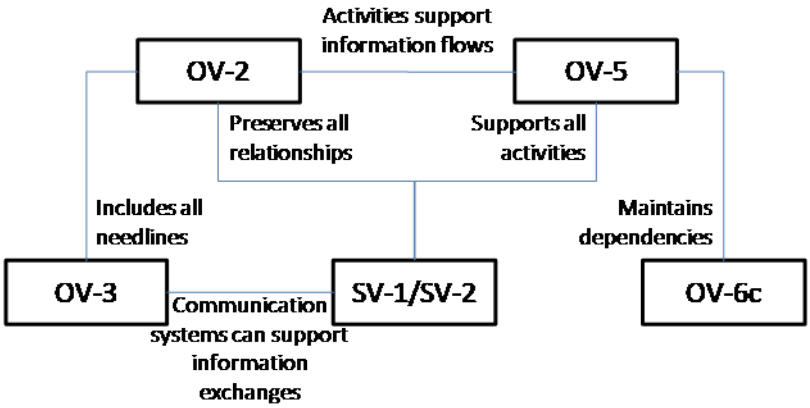


Figure 54: Relationships between DoDAF products. Reproduced from [75]

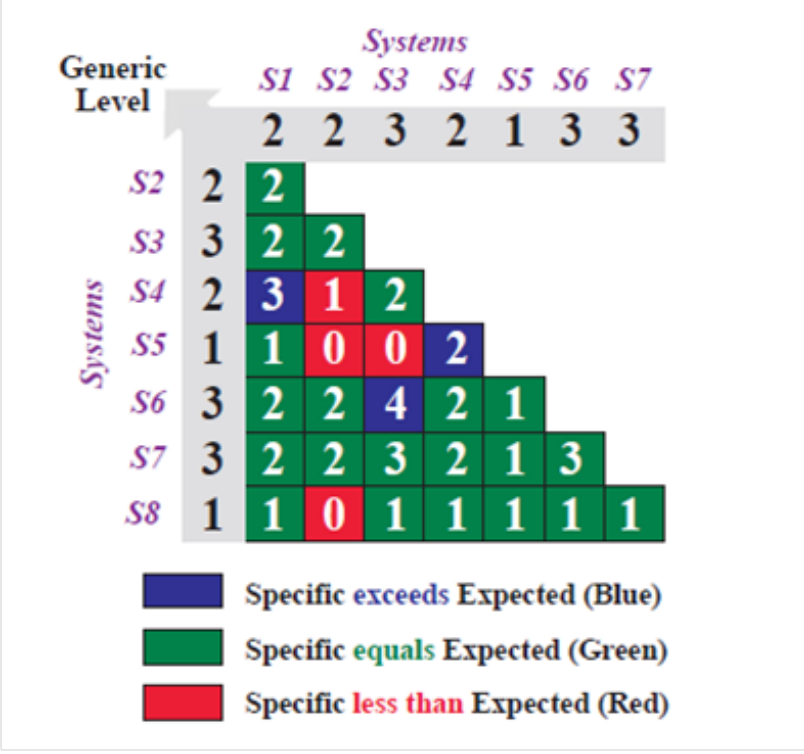


Figure 55: System-to-System Interoperability Matrix. Reproduced from [28]

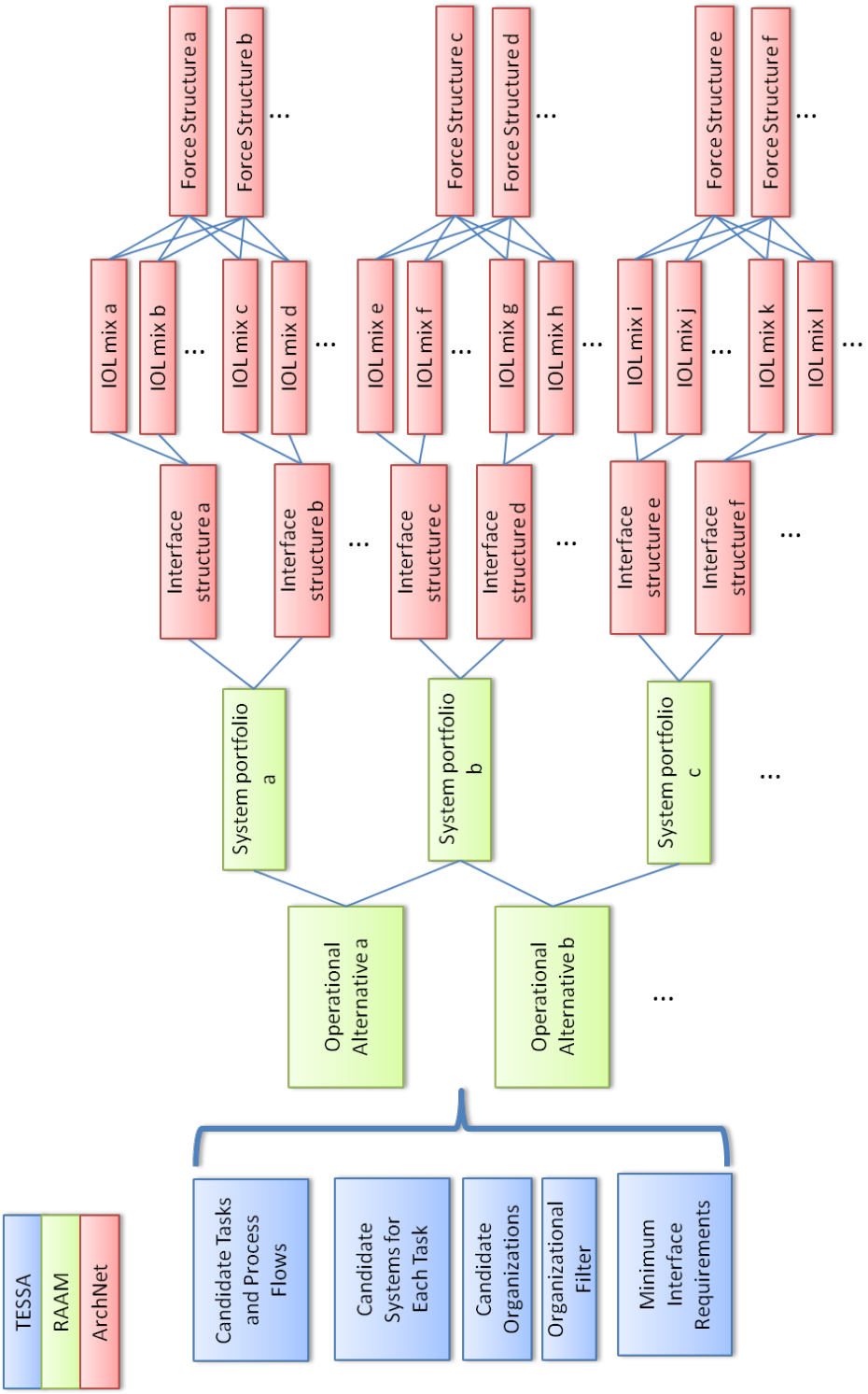


Figure 56: Full Alternative Space Description and How Alternatives are Generated

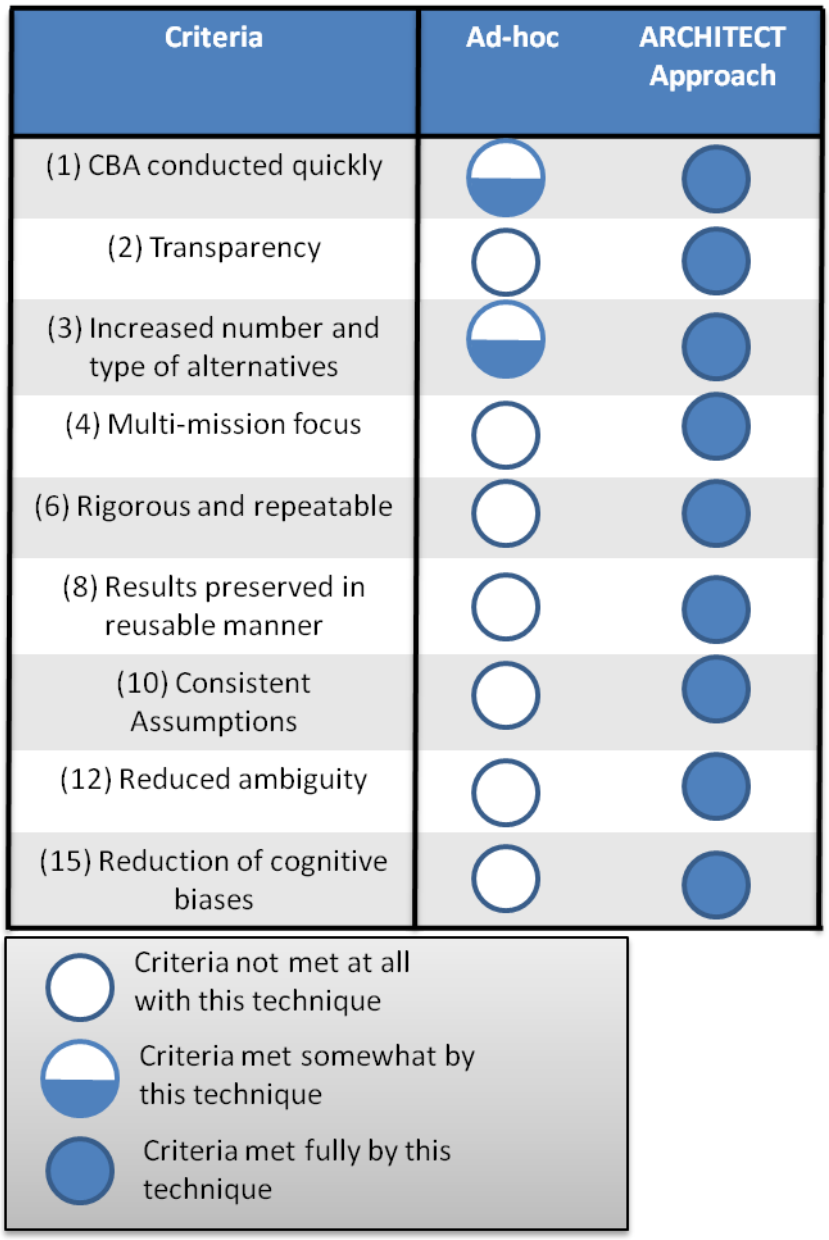


Figure 57: Qualitative Assessment of Alternative Identification and Generation Techniques Against CBA Criteria

4.2.4 Alternative Evaluation

The fifth research question deals with how to perform the alternative evaluation, and is repeated below:

- (5.1) How can quantifiable and traceable analysis of the generated architectural alternatives be performed in a timely and cost effective manner for the architecturally diverse set of alternatives considered in the early phases of defense acquisition?
- (5.2) Are the architecture descriptions provided by DoDAF products sufficient models for answering early-phase acquisition questions, what limitations does DoDAF have for executable architecting, and how can these issues be resolved?
- (5.3) What modeling tools can be used in the early phases of acquisition to provide insight into metrics of interest while using only the limited amount of information available early in the process?
- (5.4) How should information from various modeling tools be integrated to support analysis of gap fulfillment?
- (5.5) What subset of specified DoDAF products combined with what set of fit-for-use products will provide the information necessary for early-phase, acquisition-focused executable environment and how does this depend on the selection of modeling and simulation tools?

One of the biggest challenges in conducting a CBA (as discussed previously) is a lack of quantitative analysis to support decision making due to a large alternative space and the high degree of uncertainty inherent in the early phases of design and acquisition. As observed previously, executable architectures have the potential to alleviate this challenge. The technique presented here focuses on the creation of executable architectures using DoDAF to support the alternative analysis phase of the

methodology. The use of executable architecting as a key component of an early phase acquisition process can help to alleviate several of the challenges associated with architecture evaluation. An executable architecture takes advantage of the architecture views and the large amount of information contained in those views to automatically or semi-automatically generate inputs to a dynamic modeling and simulation software [69].

It has already been observed that there is a desire to increase the use of quantitative analyses in CBA. However, there are several challenges with performing quantitative analysis. First, the sheer number of alternatives being generated by the ARCHITECT method present a computational challenge for most modeling techniques. The diverse nature of the alternative space and the various dimensions of the problem to be varied make the number of alternatives exponentially increase, as was demonstrated in Figure 56. Furthermore, since the alternative space is discrete, techniques such as design of experiments and surrogate modeling to reduce the number of cases to be run will often not apply. The time required to execute a single modeling run on each alternative is prohibitive, much less the large number of repetitions required by the stochastic techniques. The exponential growth of time needed to run cases as the number of needed runs increases and the time per run increases is shown in Figure 58. Since it is often necessary to execute more than one code per alternative, or more than one case per alternative per code, the number of runs required can easily balloon. As can be seen from the figure, when the number of alternatives reaches the millions or billions range, as is the case for a full exploration of the DOTMLPF alternative space, an analysis code is required that can run cases in a fraction of a second. It is also necessary that this code be able to estimate as many of the metrics as possible, as the need for additional codes will lead to additional computational expense. These challenges will need to be mitigated in the alternative evaluation step of the ARCHITECT method.

Table 8: Modeling Techniques Assessed Against of Criteria of Interest for ARCHITECT

	Mathematical Graphs	Markov Chains	Discrete Event Simulation	System Dynamics	Engagement Modeling	RAAM	ARCNET
Execution Time	Low	Low	Medium to High	Medium	High	Very Low	Low
Information Requirements	Low	Low	Variable	Variable	High	Low	Low
Effort to Automate	Low	Low	Medium	Medium	High	Low	Low
Maximum Fidelity	Low	Low	Medium	Medium	High	Low	Low

Categorization and Comparison of Modeling Techniques for Inclusion in ARCHITECT

In order to determine what modeling tools can be used in the early phases of acquisition to provide insight into metrics of interest while using only the limited amount of information available early in the process, it is necessary to develop a taxonomy for comparing modeling techniques and assessing their appropriateness for inclusion in ARCHITECT. In 3.5, a number of existing modeling techniques that have been used for SoS analysis were presented. Each of these techniques has been assessed against the modeling characteristics presented in 3.5.1. Figure 59 shows how the modeling techniques fit into the modeling characterization tree. In addition, each technique has been assessed against the criteria presented in 3.5.1, and a summary of this assessment is provided in Table 8. These assessments are based on what has been described in the literature, and are meant to be a general qualitative assessment to help determine which modeling techniques are most appropriate to ARCHITECT and on which types of problems individual modeling types should be applied.

For each modeling type, DoDAF products that can provide some or all of the necessary inputs to automate the generation of the models have also been identified.

Seconds/Run	1	5	10	60	120	240
# of Runs	Years to Run					
1	3.17E-08	1.59E-07	3.17E-07	1.90E-06	3.81E-06	7.61E-06
10	3.17E-07	1.59E-06	3.17E-06	1.90E-05	3.81E-05	7.61E-05
100	3.17E-06	1.59E-05	3.17E-05	1.90E-04	3.81E-04	7.61E-04
1,000	3.17E-05	1.59E-04	3.17E-04	1.90E-03	3.81E-03	0.01
10,000	3.17E-04	1.59E-03	3.17E-03	0.02	0.04	0.08
100,000	3.17E-03	0.02	0.03	0.19	0.38	0.76
1,000,000	0.03	0.16	0.32	1.90	3.81	7.61
10,000,000	0.32	1.59	3.17	19.03	38.05	76.10
100,000,000	3.17	15.85	31.71	190.26	380.52	761.04
1,000,000,000	31.71	158.55	317.10	1,902.59	3,805.18	7,610.35

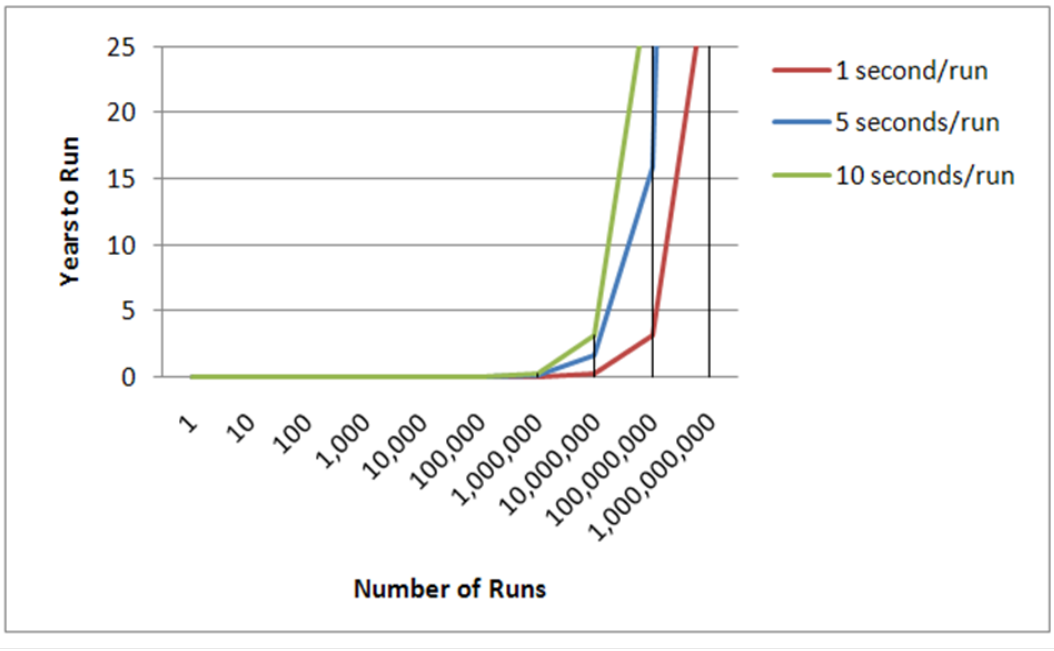


Figure 58: Run Time Based on Number of Cases and Time per Case

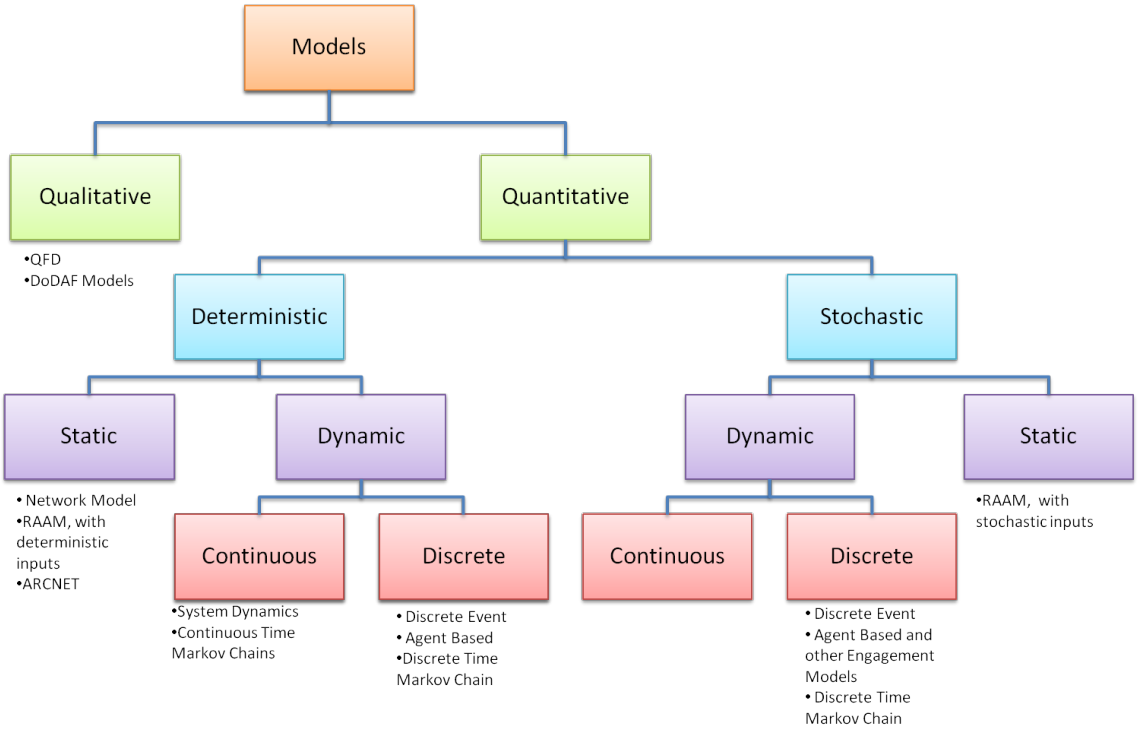


Figure 59: Modeling Techniques Mapped Against Model Characterization Tree

It should be noted that this list of products is largely in common with that required to do alternative identification, and thus the information contained in the products should be available from the alternative identification stage of the method. However, in no cases does the qualitative DoDAF models contain any of the needed quantitative inputs, and thus these will have to be gathered as an input to the alternative evaluation step of the methodology. Additionally, it it important to understand the types of information that each model can provide, so that these models can be mapped to the information needs specified in the metrics decomposition step. Based on the research performed in the literature search, Table 9 summarizes the inputs, source of inputs, and general type of outputs for each modeling framework.

Use of a Hierarchical Downselection Approach to Mitigate Challenges with Run Time and Number of Cases

Table 9: Modeling Inputs and Outputs

Required Inputs		Model Type	Types of Outputs
Input	**Source**		
Adjacency Matrix	SV-1/SV-2	Mathematical Graphs	Network effects
Edge weights	user defined		Flow estimations
States and Transitions	OV-5	Markov Chains	Distribution of time to complete
Transition Probabilities or Rates	user defined or OV-6		Probability of reaching some state
Servers and Entities	OV-5, SV-5	Discrete Event Simulation	Behavior over time
Processing time distributions	user defined or OV-6		Distribution of time to complete
Sources and sinks	SV-4	System Dynamics	Lags and bottlenecks
Flow rates	user defined		Time and amount of resource/ information flows
Activity flows	OV-5	Engagement Models	Distributions of success and loss rates
System to activity mapping	SV-5		Lags and bottlenecks
Interface rules	SV-1/SV-2		Distribution of time to complete
System performance properties	user defined		Emergent behavior
Task hierarchy	OV-5	RAAM	Any metric that can be calculated though aggregations or transformation functions
System to task mapping	SV-5		
System performance estimates	user defined		
Aggregation functions	user defined		
Required Interfaces	OV-5, SV-1/SV-2	ARCNET	Knowledge as a function of collaboration
System IOLs	user defined		
Impact of uncertainty	user defined		

The previous step of the methodology gave users the ability to generate the full set of alternatives. However, even with techniques like RAAM that could perform multiple-metric evaluations on all of those alternatives in a relatively short amount of time, it would not be possible to store all of the data on a single computer, or to visualize all of the data at the same time to allow decision-makers to compare and contrast alternatives and eliminate those that are undesirable. In fact, in a test execution of the RAAM environment on a SEAD mission, it was found that the operational and system alternative space alone (with no consideration of the additional alternatives created by interface and force structure considerations) contained over 700,000,000 alternatives [87]. One way to overcome this would be to simply place filters on the different metrics within RAAM, and save only those alternatives that meet or exceed these filters. However, there may be a more structured and methodical way to approach the issue. If branches of the alternative tree can be somehow eliminated based on low-fidelity modeling of a subset of the aspects of the problem, it would provide a means to quickly cut down the alternative space through a series of hierarchical downselections, where with each downselection, the modeling fidelity increases but the number of alternatives that must be modeled decreases. In fact, Ahmed et. al. [5] have shown that experienced designers perform more interim evaluations and downselections than novice designers, thus reducing the time and cost of the design process. It is hoped that a similar effect will occur here, where an increased number of intermediate evaluations and downselections will result in an overall increase in efficiency in the decision process.

RAAM has the ability to group alternatives by the portfolio of systems that are included in an alternative, and return the distribution of the performance of that system portfolio across all operational alternatives for that portfolio, and all task-system pairing options for that portfolio with each operational alternative. Using this information, it would then be possible to examine the system portfolios side by

side, and eliminate those that will never perform as well, or that will always cost too much, etc. Doing this would eliminate entire branches of the tree shown in Figure 56. This would then eliminate the need to even generate the rest of the descriptions of the alternatives further out on the branch, and, more importantly, eliminate large sections of the alternative space from needing to be evaluated with modeling. However, if no system portfolios are shown to be able to be eliminated based on the initial runs of RAAM, it would still be possible to eliminate specific alternatives that did not meet performance thresholds and reduce the remaining alternative space in this way.

Even after eliminating a number of system portfolios, it is likely that there would still be an extremely large number of unexplored alternatives remaining in the alternative space. In fact, even if only one system portfolio remained there would still be a large number of alternatives stemming from the different operational applications of those systems, the different ways and IOLs at which the systems could be interfaced, and the different force compositions that could be used. Thus, it will be necessary to perform further downselections. For the remaining system portfolios, RAAM can be rerun to obtain alternative assessments for all of the different operational deployments of those system portfolios, again without regard to additional alternatives created by interfacing and force structure. Filters can then be applied to this subset of alternatives based on the metrics output from RAAM to further downselect the alternative space and carry only those operational implementations of each portfolio that appear to be promising based on the analysis done with RAAM. This will further reduce the size of the alternative space and eliminate many of the interface and force structure alternatives before they actually need to be evaluated individually.

After these two initial downselections have been made using RAAM, it will be necessary to begin looking at other aspects of the alternative space, and using other types of models to increase the number of metrics that can be explored and increase the

fidelity of the estimations of these metrics. For the remaining operational-system alternatives, the rest of the alternative space is then populated using ARCNET, which can also give some initial insights into which network architectures are the most promising. With the information from ARCNET, more detailed model can then be developed to evaluate remaining alternatives. Depending on the amount of downselection that has been done prior to this step, it may be necessary to have several more iterations of increasing model fidelity and down-selecting, or it may be possible to do it in a single round. It is very difficult to create a catch-all modeling strategy at this stage of downselection, as the metrics needed and the type of problem at hand will dictate what types of models need to be created at this next level of fidelity. However, the need for an overall hierarchical approach is clear from the description of the alternative space, and has been recognized by the Air Force in their attempt to create early-phase systems engineering guidance [167].

Discussion of the Estimation of the Inputs to RAAM and Other Models

In order to enable the use of RAAM or any other quantitative modeling techniques discussed here, it is necessary to have some estimations of performance for each of the systems. For RAAM, a combination of estimates on task-independent and task-dependent metrics are required. For existing systems being used in traditional roles, it is likely that estimates of performance would be available from historical data, previous studies, or from SMEs. However, when considering new systems or existing systems used in non-traditional roles, these estimations may be more difficult to obtain.

When using SMEs to elicit performance estimations, it is important to maintain rigor in the process of eliciting data, and perform verification to the degree possible. Mathieson provides several suggestions on improving rigor in SME studies. His

suggestions include the use of clear metrics with definable and meaningful scales, a proper structure for assessment hierarchies which does not assume linear behavior for non-linear relationships, validation of SMEs as knowledgeable in the subject area and recording of their credentials, taking care to ask legitimate questions that are not outside the area of expertise for a SME, accounting for interdependencies between categories or metrics, and use of mathematical treatment of uncertainty [116]. It should be noted that the ROSETTA framework has accounted for many of these considerations by nature of its design, including no assumptions of linear behavior, accounting for interdependencies, and being able to handle and propagate uncertainty mathematically.

Ir order to propagate uncertainty in SME estimations however, one must be able to estimate the levels of uncertainty inherent in SME judgments. Kadane and Wolfson even go so far as to state, "The goal of elicitation, as we see it, is to make it as easy as possible for subject-matter experts to tell us what they believe, in probabilistic terms, while reducing how much they need to know about probability theory to do so" [104]. They go on to identify several qualities of successful elicitation, according to an overview of statistical literature. This list of qualities as to how elicitation should be carried out is repeated below [104]:

(a) expert opinion is the most worthwhile to elicit;

(b) experts should be asked to assess only observable quantities, conditioning only on the covariates (which are also observable) or other observable quantities;

(c) experts should not be asked to estimate moments of a distribution (except possibly the first moment); they should be asked to assess quantiles or probabilities of the predictive distribution;

(d) frequent feed-back should be given to the expert during the elicitation

process;

(e) experts should be asked to give assessments both unconditionally and conditionally on hypothetical observed data.

While it is generally outside the scope of this thesis to provide a full methodology for rigorous SME elicitation practices, the importance of using best practices and making every effort to reduce biases and improve the quality of estimations is recognized. Thus, it is strongly recommended that when using SME opinion in support of the ARCHITECT methodology, every effort is made to ensure that SME elicitation is done as rigorously as possible.

It is a common misconception, however, that SMEs will necessarily provide a faster and less expensive means to data collection. Mathieson [116] has suggested that the use of a judgment panel of 10-15 SMEs on an operations study can cost £10,000 in manpower costs alone, and can take months to pull together the meeting given scheduling constraints and the fact that SMEs are generally busy people. Furthermore, he suggests that two or more of these meetings are generally required to get reliable judgments. Thus, while SME interviews are one option for obtaining required information, it may be of interest to explore alternative means of obtaining this information. In addition to these concerns, there are situations in which no SMEs exist for a particular topic area, in which case other means of performing estimations of required data are needed.

One alternative would be to make small scale, higher fidelity models (such as an agent-based modes) of a system performing a specific task, and use this to help estimate the range of performance of that system on that task under a range of conditions. Since there is a relatively small number of potential system-task pairing for which no historical data would be available, and since these smaller models would be able to run relatively quickly, it would most likely be possible to use this approach to help fill in gaps in the data. Furthermore, this would be a one-time investment, as these

models and data sets could be reused in future waves. Finally, if it was necessary to create an agent-based model later on to help validate the results of the ARCHITECT method, these building blocks would be available as starting point for the creation of this model. The challenge to this approach is that these estimates would not include any potential effects of collaborations and interactions with other elements of the SoS. While this may be useful for first cut estimations made for RAAM, it would be necessary later on to use something like ARCNET to adjust these estimations for collaboration effects.

Summary of Alternative Evaluation Approach

The overall alternative evaluation approach taken here is hierarchical. As a first cut, it should be verified that all the candidate system-task mappings meet organizational constraints. This will serve to eliminate options that are not feasible for organizational reasons before ever beginning the modeling process. Next, since the alternative space is likely to still be very large (on the order of hundreds of millions or even billions of alternatives), RAAM is used in one of two ways. In the first approach, two levels of downselections are done with RAAM, one to eliminate system portfolios and another to eliminate operational uses of those system portfolios that do not meet basic performance needs. Alternatively, a blanket application of filtering may be applied to keep those system-operational alternatives which meet basic performance needs, or alternatively, keep the top performing x percent of the alternatives. This would be equivalent to attempting to keep the multi-dimensional Pareto frontier. RAAM is selected for these initial downselections because of its ability to both generate and evaluate the alternative space, and because it is orders of magnitude faster than other available modeling techniques [86]. Once this initial filtering is done, the remaining operational-system alternatives are expanded to the full alternatives

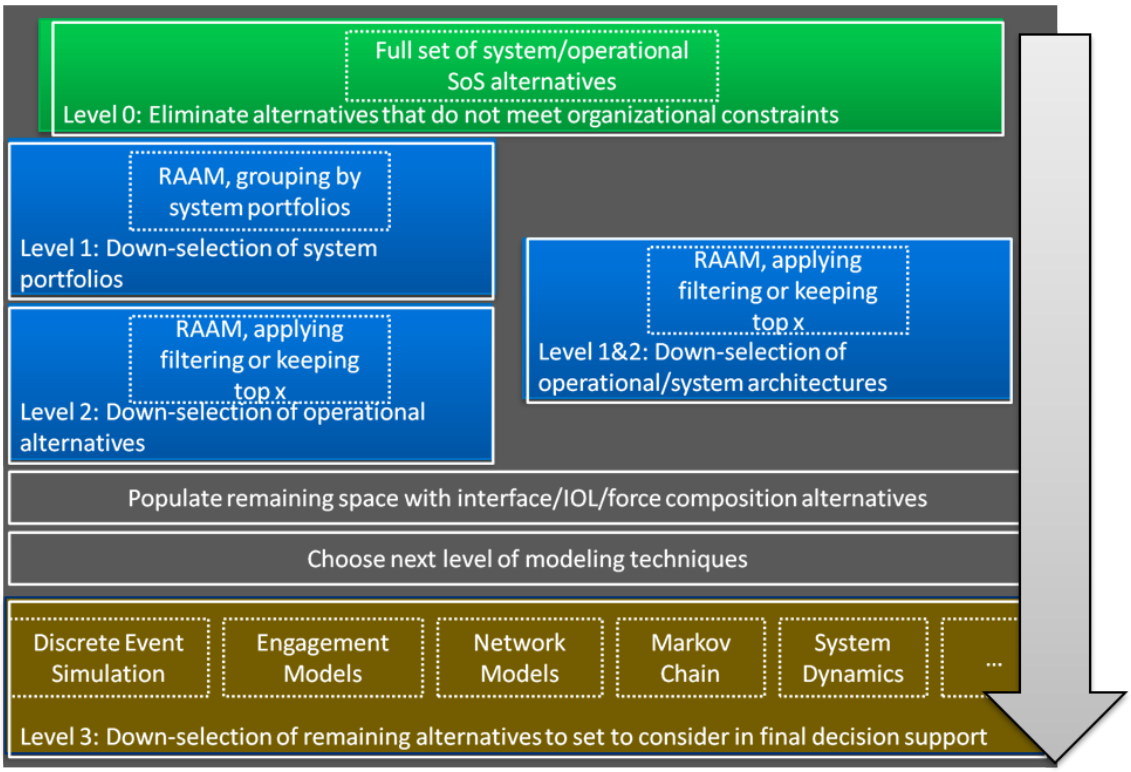


Figure 60: General Alternative Evaluation Process for ARCHITECT

space including interfaces, IOLs, and force compositions. These remaining alternatives are then evaluated using one or more of the remaining higher-fidelity modeling techniques. Which of these techniques is selected will depend on the problem at hand and the metrics that need to be calculated. However, Tables 9 and 8 can be used to aid users in determining the right mix of modeling tools. This final downselection can be used to choose that set of alternatives to be considered in the final decision support environment. At each stage, necessary data to support the creation of the executable architecture models is collected according the guidelines discussed above. This general process is depicted in Figure 60.

Notes on Stochastic Modeling

In order to help ensure that modeling is done in a rigorous way, modeling best practices must be used. While it is expected that modeling experts would be brought

in to perform modeling, and that these experts would adhere to best practices, the issue of stochastic modeling is of particular importance in SoS modeling and its impact on conducting alternative evaluation will be discussed briefly here.

Stochastic simulation techniques are often applicable in SoS because of the high-levels of uncertainty and the inclusion of human decision-making as part of the system being modeled. Stochastic modeling techniques, as a whole, rely on sampling from distributions of random variables. As such, a stochastic simulation, for a given set of inputs, does not yield the same results each time it is run. Thus, it is necessary to perform repetitions of each case to ensure that the range of possible behavior is captured. The number of repetitions that should be run is highly dependent on the model itself. If the distributions of the responses are normal, a t-test can be used to verify that enough repetitions are run to achieve a specified level of certainty on the estimation of the mean. Any introductory probability and statistics text would provide guidance on performing a t-test, and thus the implementation of the t-test will not be discussed here. Unfortunately, there is not an equivalent test for non-normal distributions. However, as a general rule, the smaller the variance, the smaller the sample size needed obtain the required level of accuracy. Conversely, this means that as variance increases, so does the number of samples (repetitions) that need to be taken. While no technique exists to ensure that an adequate number of repetitions are run for non-normal distributions, there are several ways to estimate it. Bootstrapping is a re-sampling method that uses a secondary sample taken, with replacement, from the modeling results in order to estimate the underlying distributions of the modeling results. It can be used to both estimate the shape of the resulting distribution from the model, and can be used to help estimate when enough repetitions have been run. Details on the implementation can be found in most Probability and Statistics textbooks, and will not be discussed in detail here [85].

Thus, when choosing to use a stochastic modeling approach for alternative evaluation, users of the ARCHITECT method should be aware of the implications to the total number of runs, and therefore increase in computational resources, that is necessary. This does not mean that stochastic modeling should not be used. In many cases when modeling an SoS, the behavior of the SoS is stochastic and thus it is necessary that the models capture the range of possible performance outcomes resulting from this stochasticity. However, users need to be aware of the implications of the stochastic model, and plan the modeling process and time lines accordingly.

Assessment of Architecture Evaluation Approach against CBA Criteria

It is more difficult to compare the architecture evaluation approach taken here against the CBA criteria, since the specific approach taken (particularly with respect to modeling types and data collection) is dependent on the problem being considered, and therefore there is more potential for variability in the implementation than in previous steps. With respect to conducting the CBA quickly, the evaluation approach presented here would certainly be less time consuming than using full-scale constructive simulations and wargames as has been suggested by some CBA guidance [167]. Furthermore, the number of alternatives that can be quantitatively evaluated in a given amount of time is greater than in previous CBA efforts. While it is expected that the alternative evaluation step will be the most time intensive step of the ARCHITECT method, it is also not expected to cause the execution of the full method to exceed a year, which was the goal for this criteria. Based on the initial studies presented in the following chapters, it is expected that the evaluation process will take no more than four months, depending on the size of the original problem, the number of models developed for the third downselection level, and the fidelity with which those models are created. Four months is intended to be a conservative estimate. In the examples presented in the remaining chapters of this thesis, the

alternative evaluation was completed in two months or less, including the model development time. The alternative evaluation process recommended here contributes greatly to the transparency of the overall method, as all assumptions are documented and analysis outputs are kept in data tables so that they can be revisited. Furthermore, all of the alternatives are evaluated using the same modeling techniques with the same assumptions, which allows for more direct comparisons, reduced ambiguity, and increased confidence in the comparisons. As all of the techniques discussed here can be applied across a broad range of mission types, it is possible to have a multi-mission focus. The approaches presented here use quantitative analysis, and because assumptions, equations, and inputs must be documented as part of the creation of the input files for RAAM, the modeling is repeatable. In all cases, the results are preserved in the output files and can be reused in future waves, when applicable. The hierarchical approach ensures that the full alternative space is modeling, with increasing fidelity in identified regions of interest. Because the modeling approaches taken here consider the whole SoS, some attention must be paid as to how any new system or use of a system will integrate within the SoS. However, a full study on integration requirements is not completed as part of this process. While it is important to think about how difficult a solution may to be to integrate, and initial estimates of integration difficulty should be reflected in the SoS modeling, there is not enough information available during early phase acquisition to perform a full analysis of the integration. The most challenging criteria is the criteria to verify the feasibility of the expected outcomes, and that the solution will integrate into the existing architecture. While this method pays more attention to integration than is typically paid in CBA studies, it is very difficult to account for the 'unknown unknowns' in the SoS. In fact, verification and validation is one of the most challenging issues in SoSE, since there is so rarely historical data or full-scale prototype testing on which to base verification and validation. Thus, while every effort is made in the method to ensure that the

anticipated outcomes are feasible upon integration, it is impossible to ever fully verify the expected results until prototype testing in done later in the process.

With respect to the reduction of cognitive biases, there are three potential biases which were identified for alternative evaluation and selection. The first bias, representativeness, results from an over-estimation of the degree to which a case study or example represents the problem at hand. Since models are made for the specific scenarios and missions being addressed, it is expected that this should not be an issue. Rather than trying to draw conclusions from similar case studies and examples, the actual case itself is modeled with increasing fidelity, and thus is expected that the results are applicable to the problem at hand. The second potential bias, illusion of control occurs when decision-makers overestimate their personal influence and ability to mitigate risks of a potential outcome. This bias is more applicable to the decision support step of the ARCHITECT methodology. However, improving the overall quality of the supporting data and performance predictions for each should contribute to increased transparency with regard to risks, and should increase the overall awareness of the levels of uncertainty and the sources of these uncertainties. The third bias of interest is the devaluation of partially described alternatives. Since the ARCHITECT process requires that all alternatives be described at the same level and evaluated in the same way, all alternatives are being compared with the same level of description for each, effectively eliminating this potential bias. The assessment of the ARCHITECT approach as compared to both ad-hoc approaches and constructive simulation and wargaming approaches against the CBA criteria is summarized in Figure 61.

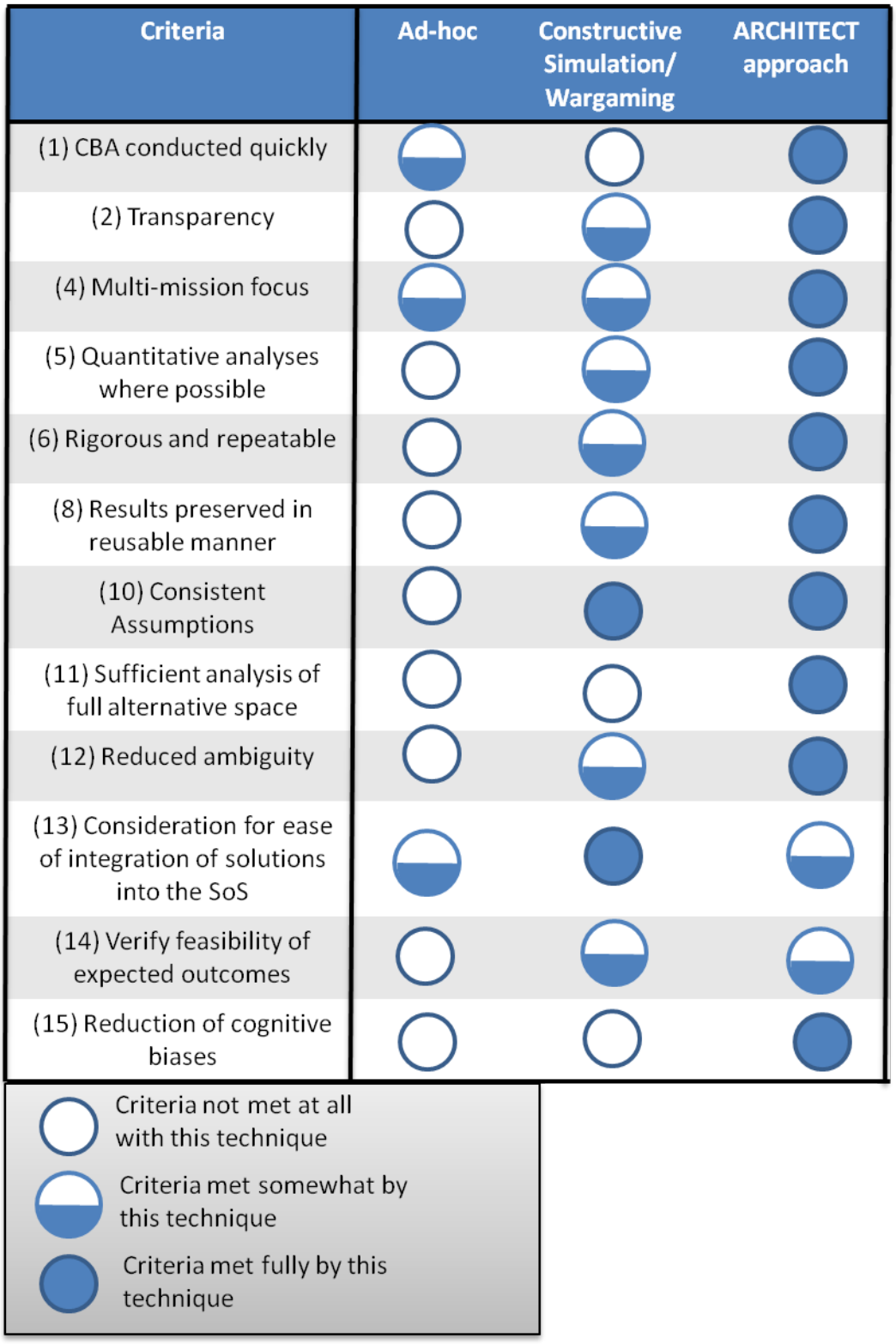


Figure 61: Qualitative Assessment of Alternative Evaluation Techniques Against CBA Criteria

4.2.5 Decision Support

The sixth and final research question deals with the area of decision support, and was: How should be a decision-support environment to improve early-phase acquisition decision-making be developed, and what should be the features of such an environment? Although decision-support is important to the ARCHITECT methodology, it is perhaps one of the more well-developed areas considered as part of this research and has not been a focus in the initial development of the ARCHITECT methodology since many of the other areas were lacking in established techniques and approaches that were appropriate to the analysis of SoS in a CBA context. Therefore, some overall recommendations and guiding principles will be made based on techniques that have been found in the literature which are particularly applicable, but a formal approach for decision-support for ARCHITECT will not be developed as part of this thesis. Future work should expand this area and add additional formalism to the method.

As decisions are required throughout the execution of the ARCHITECT methodology, decision support is needed at all stages of the process, culminating in the final decision support environment to enable pre-Milestone A decision-making. Therefore, rather than being a single step of the methodology, it is really an ongoing process where decisions are required at each step, and sound decision-making principles should be applied throughout the methodology. Thus, the decision support is not so much a step of the process as something that is intertwined with the process and must be practiced with every step along the way. Instead of a single decision support environment that is created at the end of the process, a decision support environment is needed that helps the decision-makers wade through all of the steps of the methodology, and acts to record all information and decision rationale at each step of the process to increase the transparency.

A reusable ARCHITECT environment which aids the execution of the ARCHITECT methodology is ultimately desirable, but as the creation of this environment is a software engineering challenge and outside of the scope of this thesis, a roadmap of such an environment will instead be laid out here. An initial prototype of the ARCHITECT environment has been developed along with one of the example problems, but is still in progress and is not a contribution of this thesis. The development of the requirements of this environment will instead be a contribution of this thesis, with the full implementation of that environment being left to future work.

In addition to the overall decision-support process needing to meet the CBA criteria, the environment will need to meet the criteria laid out by Ullman, which include (with some omissions) the ability to support inconsistent decision-making information, the ability to support incomplete decision-making information, the ability to support uncertain decision-making information, the ability to support evolving decision-making information, the ability to support the building of a shared vision, the ability to suggest direction for additional work and what to do next, the requirement of an overall low cognitive load, the ability to support a rational strategy, and finally, the decision support environment should leave a traceable logic trail, which is redundant with the CBA requirement for traceability.

Many of these criteria are met easily by the selection of the techniques which enable the previous steps of the methodology. For example, the use of ROSETTA can support inconsistent, incomplete, uncertain, and evolving information. The ability of all of the steps of the method to meet the call for traceability has been discussed in each of the step's development sections individually. It has been discussed in the development of the alternative evaluation step that information is expected to be inconsistent, incomplete, uncertain, and evolving, and the hierarchical method proposed in that section has been designed to help in these areas, as well as in the area of suggesting additional work and what to do next. In fact, not only does this method

attempt to address the issue of inconsistent and incomplete information, it attempts to rectify both of these issues to the extent possible so that inconsistent and incomplete information are not issues. The requirement for a low cognitive load will be difficult to meet because CBA by its very nature deals with complex SoS, but the environment needs to be designed such that its use will not increase the overall cognitive load. Again, the selection and development of the techniques in each step can also aid in this area. Each step has been designed to reduce the cognitive biases and need for cognitive simplifications, thus helping to address this criterion. The process has also been automated to a high degree in order to lower the cognitive load on decision-makers and SMEs and reduce user input requirements. Since the information is all contained in one place and there has been an emphasis on consistent assumptions and shared and equal data for all alternatives, the requirement for a shared vision has been met as well. The ability to support the development of a rational strategy focuses on spending time upfront developing both clear goals and criteria for meeting those goals, which has been shown to improve the overall quality of the final decisions. ROSETTA is designed to do this. It leverages and expands on QFD, which is designed to help better capture and understand requirements and map them to criteria which can be used by engineers to evaluate the pros and cons of design alternatives. ROSETTA takes this a step farther by improving the mathematics embedded in the technique and by having the ability to double as a framework in which to evaluate potential solutions against the requirements.

While all of these criteria are met by the techniques selected to enable the ARCHITECT method, the environment must also accomplish these things visually, and also must meet the needs of the CBA criteria. In addition to the transparency requirement already discussed, it is also desirable that the decision support environment help the CBA to be conducted quickly, be able to support a multi-mission focus, enable rigorous and repeatable data analysis, be dynamic, preserve the results and

the decision logic (again, redundant with the above requirement to leave a traceable logic trail), enable rigorous data integration, help ensure that all users have consistent assumptions, help to reduce ambiguity, help to assess the feasibility of the expected outcomes, and help to reduce cognitive biases. While many of these are again met by the techniques working within the environment (and the meeting of them has been discussed in the previous sections), the application of visual analytics in the development of the environment can be used to reduce cognitive biases and make the decision support environment more dynamic. The application of visual analytics is required to meet one criterion that is unique to the decision making element of the methodology, the need for a dynamic decision environment. JMP® will be used as the platform for visual analytics in this work, due to its availability. However, the user of the ARCHITECT method could use any visual analytic software package with which he or she is familiar and which enables the required analyses.

Although there are many places within the ARCHITECT methodology where decisions will have to be made to progress through the method, a final decision needs to be made at the end of the CBA which either results in a DOT_LPF change recommendation or recommends a technical approach (materiel solution) to carry through Milestone A. ARC-VM is selected as an enabler for this decision. ARC-VM integrates the data gathered in the previous steps with a rigorous measure of the architectural complexity of each alternative to establish an overall comparison of each alternative with respect to its overall acquisition value and uncertainty. Since the remaining alternatives at this final decision step will most likely all close gaps sufficiently, the use of ARC-VM allows decision-makers a way to compare them with respect to acquisition program considerations, thus choosing the alternative(s) most likely to both meet needs and be implemented successfully. While this is not a complete answer to the need to verify the feasibility that the solution will perform as promised once fielded, it is a step in that direction and has to the potential to meet that need with continued

testing, research, and development. Although it is recommended that ARC-VM be used to aid decision-making in the final stage of the ARCHITECT method, it does not, and should not stand alone. ARC-VM should be implemented in a visual analytics framework that allows users to interact with the architectures and delve into the data for each of the alternatives, thus fully understanding why they have been placed into a particular region of the tomato garden plot. Decision-makers should also be able to make perturbations to the architecture alternatives in this environment, and re-analyze the perturbed architecture on-the-fly. This will allow decision-makers to understand the impact of changes to the architecture and explore the impact of potential course changes or known risk areas.

4.2.6 Integrated Methodology Summary

A high-level depiction of the process employed by ARCHITECT is shown in Figure 62. Figuure 63 shows a more detailed step-by-step decomposition of the methodology. In order to facilitate the use of the method by SMEs and decision-makers, a framework has been developed. The framework has a series of interactive dashboards that guide users through the execution of the method. Each tab of the ARCHITECT framework corresponds to one workshop with either SMEs, decision-makers, or both. A diagram of the framework and the process for using the framework is shown in Figure 64.

The ARCHITECT methodology begins in the upper left-hand corner of the vee. First, missions of interest are identified and the baseline architectures are documented (or if existing, collected). This research makes the assumption that if a CBA is being conducted, it is because the person or organization requesting the study has identified a potential mission gap, or is interested in whether a proposed system can help fill capability gaps in a mission or set of missions. The methodology is designed to handle a multi-mission space, although it can be used equally well to consider only one mission. Although one mission may be selected as the focus for closing capability gaps, other

missions that may be impacted by any proposed solution are also able to be included in the study. This is particularly applicable to large scale materiel solutions which will likely be employed across several missions if deployed operationally. Missions of interest are identified by those requesting the study, although it is anticipated that SMEs may later suggest additional missions to be included in the study. Again, it is assumed that the problem definition is supplied, as this is the standard procedure for kicking off CBA analysis. However, it is possible that additional missions are added by those performing the study, and these would be added via literature search and SME consultation. For each mission, the baseline architecture and mission performance should be established, again, either as part of the problem definition, with literature search, or by consulting SMEs.

Once the mission set is established, a set of metrics is then defined that provides a means through which mission success can be quantified. In the methodology, ROSETTA has been chosen as a framework with which to perform the decomposition from the high-level mission needs to the MoE and MoPs. ROSETTA is further used to eliminate redundant metrics and reduce data collection requirements. This is combined with PSM to ensure that the resulting set of metrics meet the criteria for good metrics. For the purposes of testing the method, the INCOSE criteria for good metrics are used, which are relevant, complete, timely, unambiguous, logical, simple, cost-effective, repeatable, and accurate. Additionally, for each metric, an aggregation function is identified for use in RAAM.

A literature search is combined with the use of subject matter experts (SMEs) to estimate the relationships in ROSETTA required to identify, quantify, and rank capability gaps. Although it is possible with ROSETTA to project the gaps all the way up to the mission level, it may be more useful (and more in line with CBA guidance), to examine the gaps at the capability level, or even at the MoE level. This will provide a clear baseline for comparison of alternatives later in the process. The

baseline specification, metric derivation, and gap analysis are performed on tab 1a of the framework.

The scope of the alternative space is then defined by a combination of SMEs and decision-makers using TESSA to clearly identify the boundaries of the alternative space that the decision-makers are willing to consider. This is done by first specifying the operational alternatives through the rules for variations on the activity sequences, and then specifying the list of alternative systems available to perform each activity, their compatibilities, and estimations on their performance against all metrics of interest. For each system and activity, the owning and responsible organizations are identified respectively, as well as whether the decision-maker is willing to consider a shift in organizational responsibilities. Finally, a minimum IOL is specified across all necessary interfaces. RAAM is used to automatically generate alternatives within this scope. Alternative boundary specification is done on Tab 1b of the framework.

These alternatives are then analyzed and filtered using a multi-stage approach. First, RAAM is used to evaluate the alternatives against all metrics that can be evaluated using aggregation functions. Since it is infeasible to view, or even store, the full set of data that results from RAAM, several options are available to reduce the amount of data to be parsed for an initial down-selection. First, RAAM can be set to save only the top x performing alternatives, where x is user specified. Alternately, RAAM is able to group solutions by the system portfolio used in those alternatives, and then, for each portfolio, calculate the average and variance for each portfolio. This option is of interest in conducting a CBA, since the CBA is attempting to select between materiel and non-materiel alternatives. Using this approach, all of the system portfolios can be compared, and decision-makers can determine whether or not top-performing portfolios include new systems. Since the performance of these systems is uncertain, RAAM can use probability distributions in place of exact performance estimates. Since it is likely that the top performing alternatives will be statistically

indistinguishable, the initial RAAM modeling is used to down-select to a smaller group of portfolios that will be carried forward to the next phase of analysis. This down-selection is performed by decision-makers, using a visualization environment that is on tab 2 of the framework.

Once a smaller set of portfolios has been selected, these portfolios are re-run through RAAM, where each operational architecture associated with each portfolio is then recorded independently. This allows only the top performing operational architectures for each system portfolio to be carried forward to the higher-fidelity analysis. This secondary down-selection occurs on tab 3 of the framework.

The remaining architectures are then run through a higher fidelity modeling environment. The specific technique used in this environment is dependent on the problem at hand and the metrics being tracked. However, discrete event simulation, Markov chains, network models, and ARCNET paired with a simplified engagement model have been identified and tested as part of this thesis as techniques that can easily be incorporated into an executable framework and provide information that may of interest during the process.

Finally, an abbreviated list of alternatives that best fill capability gaps is analyzed using a decision support environment (DSE). The DSE provides an interactive, dynamic visualization environment of the relevant data and analyses to help decision makers choose the appropriate path forward. The DSE is also intended to guide additional analysis that may be required to increase confidence in the chosen solutions and to verify mission impact. Ultimately, the DSE is designed to aid DMs in leveraging the entire method when addressing important pre-Milestone A specific concerns such as: *Should a new system acquisition program be launched, are there affordable architectural alternatives that best provide the needed capability, or are there any new methods of employing existing assets that can be leveraged to achieve the same desired effect?* The DSE is located on tab 4 of the framework.

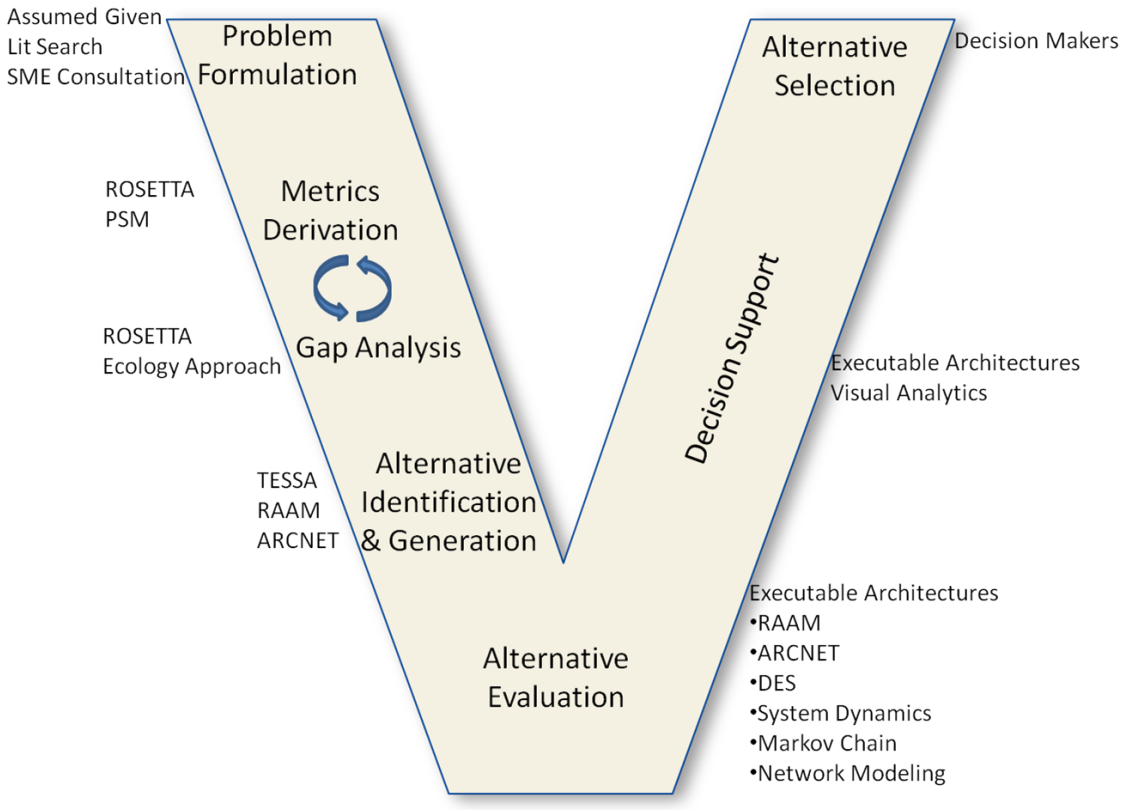


Figure 62: ARCHITECT Vee with Enablers Mapped to Methodology Steps

It is important to realize that the ARCHITECT method is iterative. If metrics are deemed missing, or a new mission needs to be added, the process is designed so that updates can be done with minimal time and effort. This will allow CBAs to be more complete while errors can be corrected without significant expenditures of time and cost. This is accomplished through a high level of automation between the steps of the process, and through the creation of a flexible environment that enables users to quickly and succinctly collect and document all required information in the appropriate format for the automation.

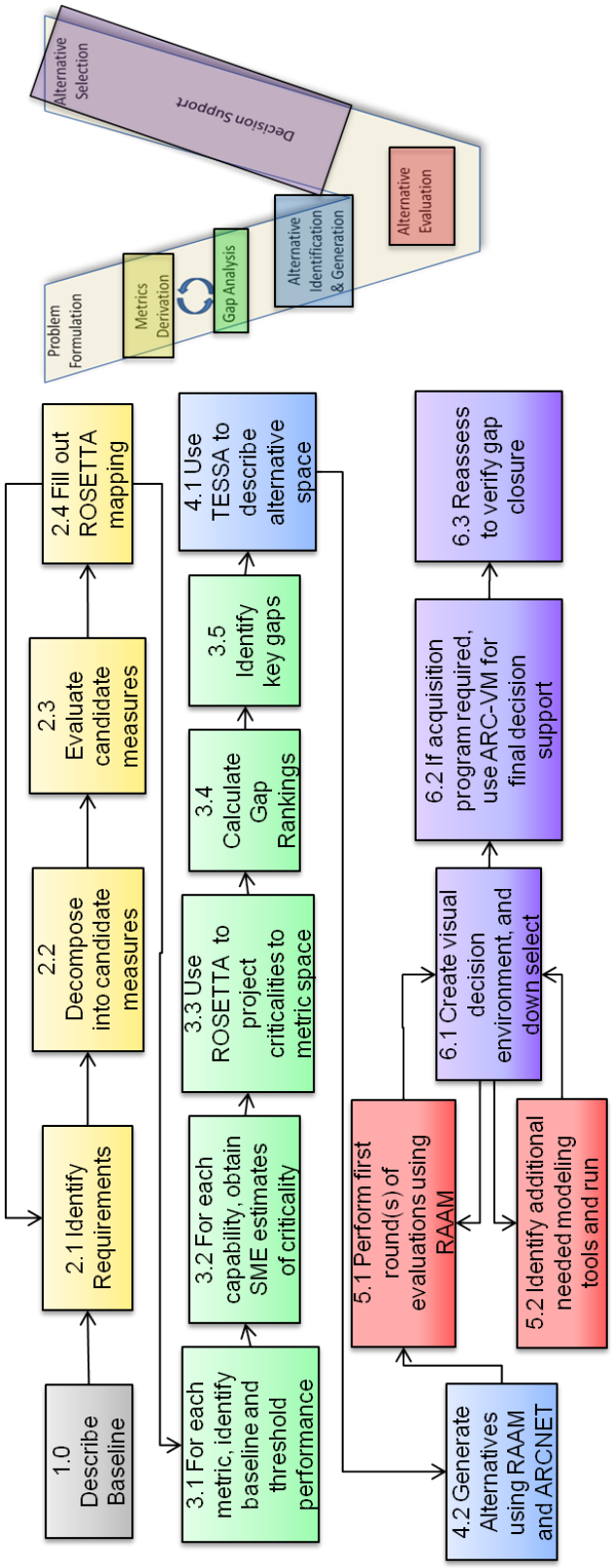


Figure 63: ARCHITECT Methodology Flowchart

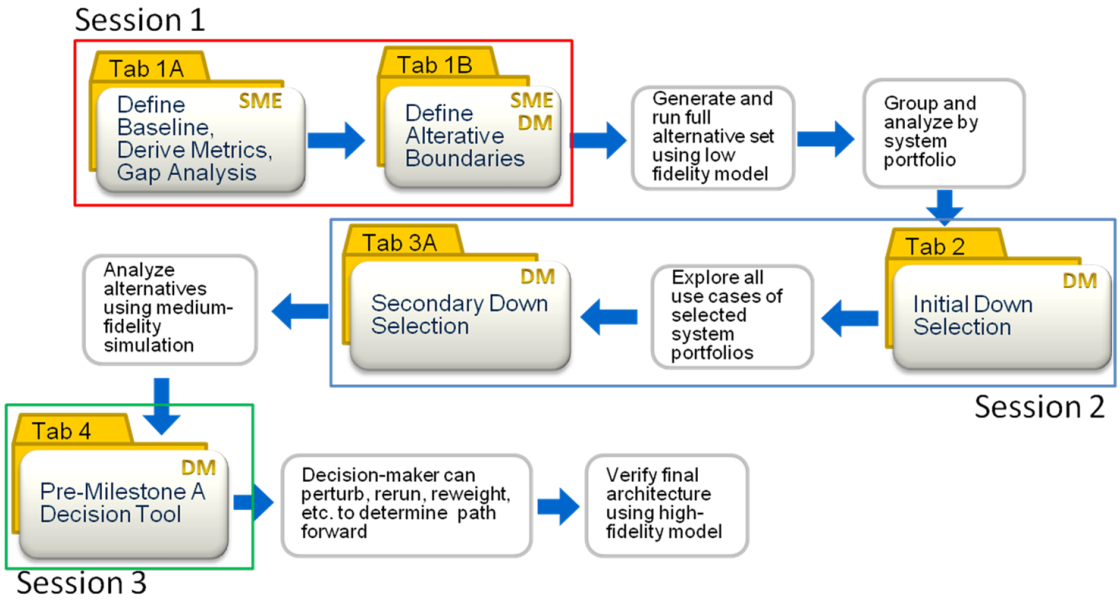


Figure 64: ARCHITECT Framework Use Flowchart

CHAPTER V

IMPLEMENTATION OF THE ARCHITECT METHOD TO A SEAD CASE STUDY

In order to perform and initial demonstration of the feasibility of the method, a suppression of enemy air defenses mission will be used. SEAD is defined as any activity that neutralizes, destroys, or temporarily degrades enemy surface-based air defenses by destructive and/or disruptive means [48]. Systems such as Surface-to-Air Missile (SAM) sites, Anti-Aircraft Artillery (AAA), early warning and fire control radars, and Ground Control Intercept (GCI) sites can be combined by potential adversaries into an Integrated Air Defense System (IADS). Over time, IADS have become increasingly complex and can differ widely in terms of organization, sophistication, and operational procedures. The widespread proliferation of weapon systems and continual improvements in their speed, range, accuracy, stealth, and lethality require joint forces to be more responsive, flexible, and integrated. Since SEAD can be conducted jointly, a multitude of various system types can be included within the architecture. These range from sea, land, air, and space-based assets to manned and unmanned systems. Also, sophisticated communication systems are needed to enable and enhance Command and Control (C2) since enemy air defenses can be mobile or stationary and pose a significant threat to current military assets. Thus, SEAD presents an excellent SoS architecture design challenge and includes many of the aforementioned attributes including: managerial and operational independence as well as geographical distribution of elements, emergent behavior, and evolutionary development. For purposes of this study, a SEAD scenario that focuses on Area of Responsibility-/Joint Operating

Area (AOR/JOA)-Wide Air Defense System Suppression provides the desired complexity for a military SoS architecture test case. This means that SEAD is conducted against specific enemy air defense systems throughout the AOR/JOA to degrade or destroy their major capabilities/effectiveness. The duration and level of disruption depends upon the mission objectives and the sophistication of the IADS [48]. In this case, it will be assumed that the SEAD mission is conducted in an area that is easily accessible by both an aircraft carrier and a forward operating base (FOB) in order to demonstrate joint and cross-service trades.

The study presented here is intended to demonstrate the plausibility of the ARCHITECT methodology. While the study presented in this paper is limited to a single mission area for ease of demonstration, the ARCHITECT methodology does allow for other SEAD mission types, such as localized and opportune suppression, as well as other mission types to be included for future analysis. The single mission focus allows a direct comparison with previous CBAs and can be used to more directly compare the ARCHITECT approach to previous studies. However, as was observed previously, one of the recommendations of ARCHITECT is to use a multi-mission focus on future CBAs. Data used in support of this study is notional and is not intended to reflect the actual performance of the real systems, again, to mitigate export control concerns and to avoid the accidental production of potentially sensitive information.

The scenario development, RAAM,and ARCNET models that are used in this chapter were developed as part of a larger research effort sponsored by ONR, and documentation of the development of the models for this research effort can be found in [4], which is the end of the year report for the sponsored project for fiscal year 2011.

5.1 Definition of Baseline

Prior to performing gap analysis, a baseline architecture must first be defined. In order to support this process, information in the form of documentation must be gathered. This documentation includes, but is not limited to documents that outline the appropriate doctrine, concept of operations (CONOPS), task lists, reports, etc. The use of this documentation is not limited to providing performance estimates of the baseline architecture, but also of the candidate alternative architectures for use in later modeling and simulation (M&S) efforts as well. Because all aspects of the SoS architecture must be considered in the alternative space to make meaningful comparisons, a subset of DoDAF v2.0 models is used to document the baseline architecture, and to help identify the scope of alternatives to be considered. The models that are used from the associated viewpoints are the following:

- Operational Viewpoint
  - OV-1: High Level Operational Concept Graphic
  - OV-2: Operational Resource Flow Description
  - OV-3: Operational Resource Flow Matrix
  - OV-4: Organizational Relationships Chart
  - OV-5a: Operational Activity Decomposition Tree
  - OV-5b: Operational Activity Model
- Systems Viewpoint
  - SV-1: Systems Interface Description
  - SV-2: Systems Resource Flow Description
  - SV-5b: Operational Activity to Systems Traceability Matrix

The OV-1 is used to document which missions are being considered and help capture any associated assumptions about the mission. It is also important to understand the organizational context of the roles and relationships amongst different stakeholders so the OV-4 is included as well. The OV-5a details the hierarchical structure of the activity sequences for the missions while the OV-5b provides contextual data to help depict the relationships among activities, inputs, outputs, performers, or other pertinent data. The OV-2 & OV-3 provide a description of the required resource flows exchanged between the operational activities. The SV-1 & SV-2 models provide the identification of systems, system items, and their interconnections as well as the resource flows exchanged between systems. The SV-5b is included to understand how the systems enable the activities shown in the OV-5, and to ensure that a set of systems selected for use in architecture is able to fully support the needs of the missions.

DoDAFv2.0 states that "There is no single, correct way to visualize any view" and that "There are multiple techniques that can be employed creating architecture models in differing views" [50]. In order to make the desired process automate-able so that large design spaces can be evaluated, these baseline views need to be created in a format that is computer readable, and standardized. This will reduce the difficulty in generating alternative architectures by allowing direct manipulation and permutation of the baseline views. The added advantage is that the same architecture views used to document and display the candidate architectures can be used as the backbone of an executable architecting environment. This approach is derived from the idea alluded to earlier that leveraging the ideas of executable architecting can help speed up this process and provide support to decision makers who may wish to consider additional alternatives prior to making a decision.

For the SEAD example, it should be noted that while the baseline was created to be representative, the data used is notional and not intended to reflect actual system

performances. Any resemblance to actual performance data is coincidental. The OV-1, OV-2, OV-3, OV-4, OV-5, SV-1, SV-2, and SV-5b were created for the baseline case. Although not all views are depicted here, the OV-2 and OV-5b with system overlays are shown to provide description of the baseline assumed for this study. In addition to the architecture views, baseline SEAD JOA/AOR performance data is presented. The formulation of this data is based on a literature search and the best estimations of the author, and again should not necessarily be considered as an actual representation of real mission data. This data is only intended to allow for the utility of the method to be demonstrated. The OV-1 is shown in Figure 65. The OV-2 is shown in Figure 66, and the corresponding OV-3 is shown in Table 10. The baseline OV-4 is shown in Figure 67. The baseline OV-5b and SV-5b is shown in Figure 68. The SV-1/2 for the baseline is shown in Figure 69. Note that the versions shown here are pictorial version created for ease of display and are not the machine readable versions used in the ARCHITECT analysis. The information used to develop this baseline was based largely off the information detailed in [177].

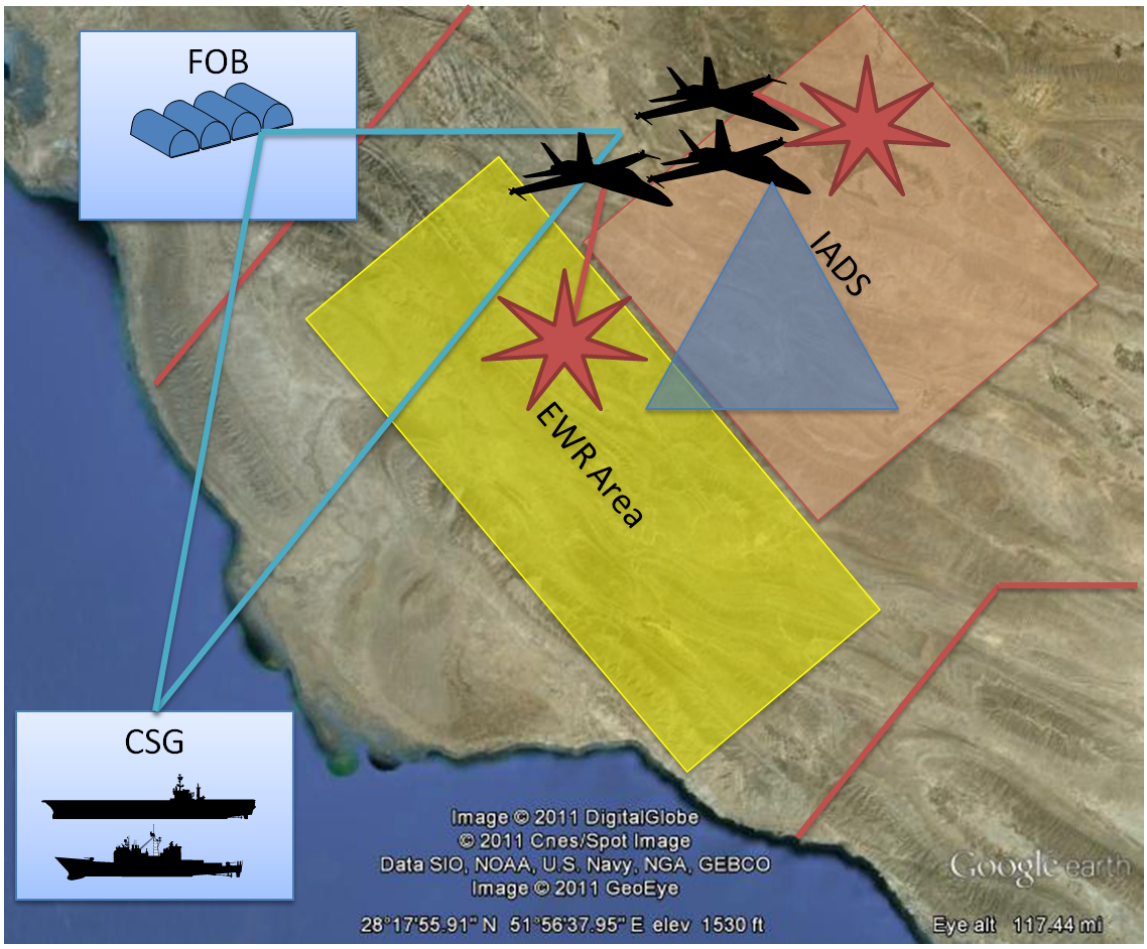


Figure 65: SEAD Baseline OV-1

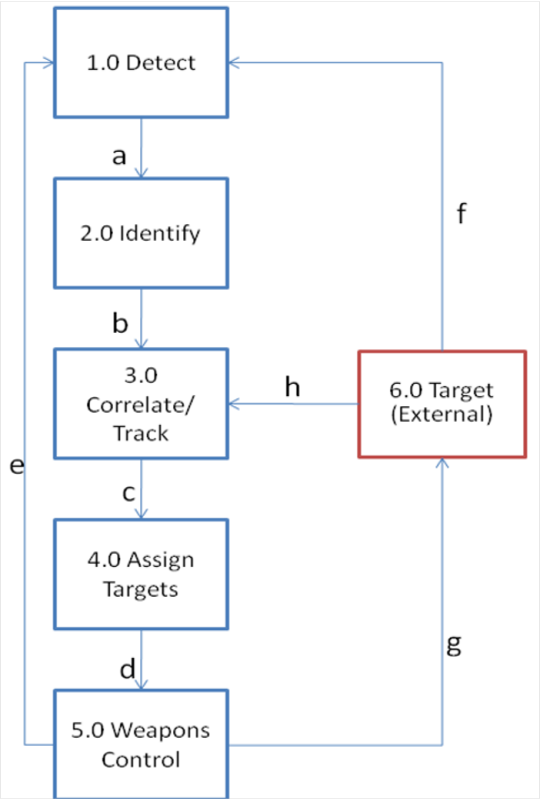


Figure 66: SEAD Baseline OV-2

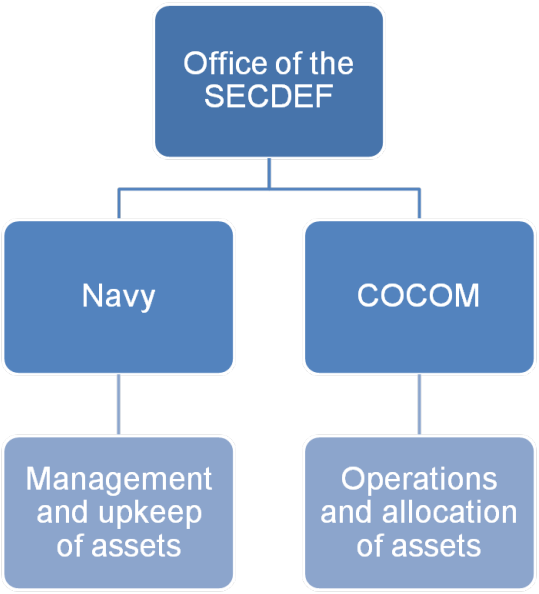


Figure 67: SEAD Baseline OV-4

Table 10: SEAD Baseline OV-3

ID	*Source Node*	*Nodal Activity*	*Resource Type*	*Details*	*Receiving Node*
a	Detect	Coordinate Sensors	Information	Warning/Location Data	Identify
b	Identify	Identify Friend from Foe	Information	Positive Enemy Target ID	Correlate/Track
c	Correlate/Track	Maintain Positive Target Track	Information	Updated Target Track	Target Assignment
d	Target Assignment	Target Assessment and Assignment	Information	Target Assignment Plan and Permission to Fire	Weapon Control
e	Weapon Control	Engage Target	Physical and/or Energy	Destructive or Disruptive Weapon	Target (External)
f	Target (External)	Sense Target	Information and/or Energy	Sensor Energy/Data	Target Assignment
g	Weapon Control	Engage Target	Information	Battle Damage Assessment	Detect

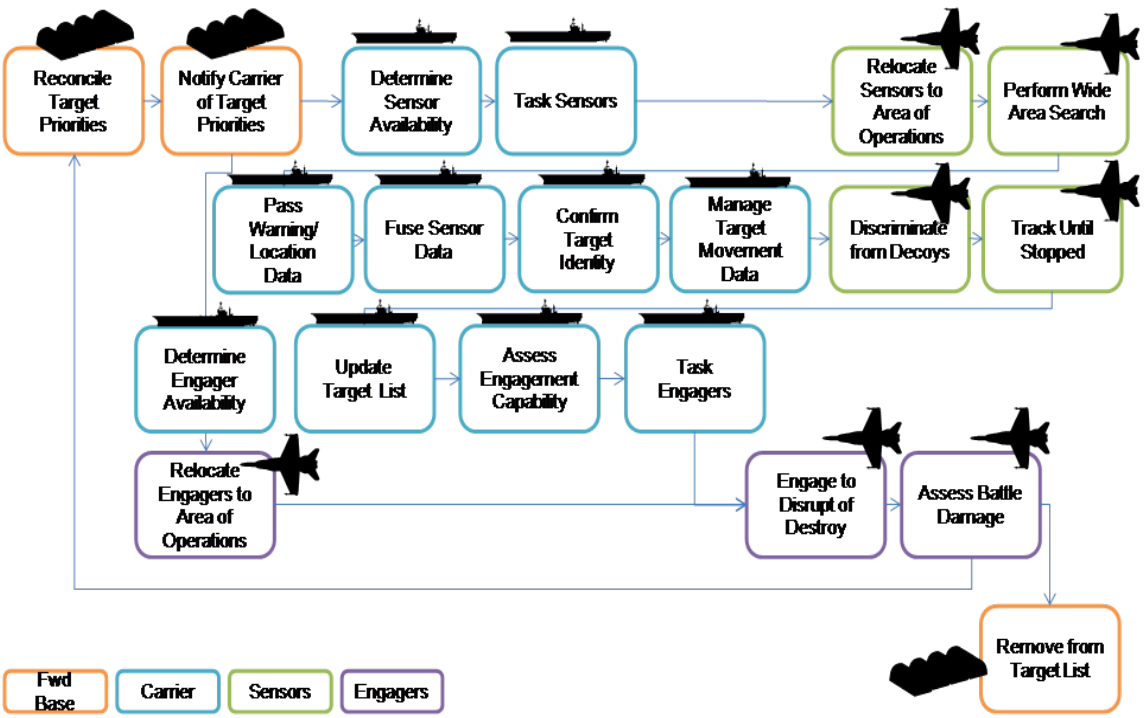


Figure 68: SEAD Baseline OV-5b and SV-5b

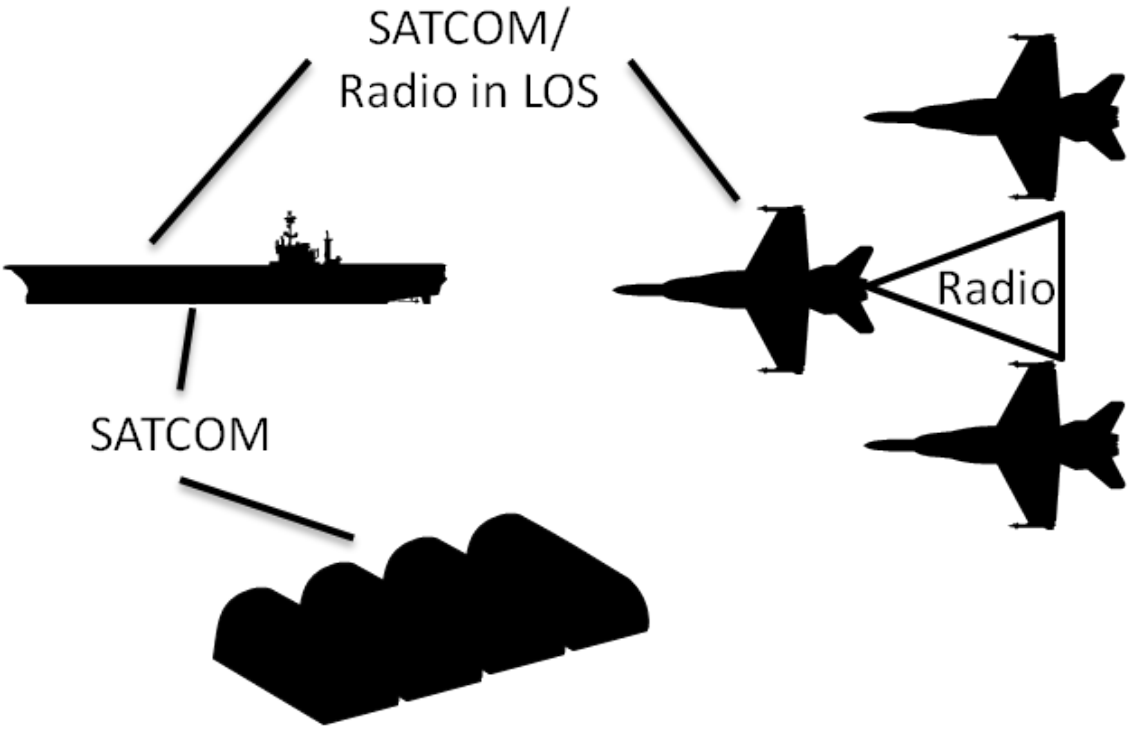


Figure 69: SEAD Baseline SV-1/2

5.2 Metrics Derivation

In order to derive the metrics, the high-level goals for SEAD JOA/AOR needed to be defined. The overall goal of SEAD is to effectively disable an enemy's air defenses in order to protect friendly aircraft flying over that area in a follow-up mission. Thus, the main goals are to effectively disable the air defenses, do so in a timely manner, do so with minimal friendly losses, and, of course, to do so in a cost-effective manner. While the first three can be measured by the percent of targets disabled, the probability of success, the time to complete mission, and the percent of friendly losses respectively, measuring the cost-effectiveness is more complicated. As there are many contributors to cost, and not all of them are likely to have available quantitative estimates during CBA, it will likely be necessary to use a mix of qualitative and quantitative assessments for cost. System acquisition costs are likely to be available for most systems, and estimates for new systems can most likely be obtained. The operations and support (O&S) costs are unlikely to be available, as well as the costs of integration into the SoS. However, based on the discussion in the literature search that complexity and cost are correlated in an SoS, the complexity measure developed by Domercant [59] will be used here as a surrogate for cost. Since this metric requires more information than is available early in the evaluation process, the complexity will need to be initially estimated qualitatively. However, as the evaluations progress and the architectures become more fully fleshed out, the complexity can be calculated using Domercant's metric. In addition, a qualitative estimation of maintainability can be obtained as well. This gives three metrics that will contribute to helping to assess the relative costs of the architectures, the acquisition costs of the systems, the complexity score, and the qualitative estimates of the maintainability. This is summarized in Table 11.

The metrics specified above are not unrelated. For example, the probability of success would be expected to increase if more targets are disabled, meaning that it

Table 11: Metrics Derivation for SEAD

	Probability of Success	% Targets Disabled	Combat Attrition (% Friendly Losses)	Time to Complete Mission	Ease of Maintainability	System Acquisition Cost	Complexity
Effectively disable air defenses	x	x					
Timely disablement of air defenses				x			
Reduce friendly losses			x				
Reduce mission costs					x	x	x

may only be necessary to track one of these two metrics. It would also be expected that an increased probability of success would be correlated with fewer friendly losses. It also might be expected that there be some positive correlation between complexity and probability of success, as more complex architectures are sometimes expected to increase mission performance. It also would be expected that the time to complete mission would be correlated with both the percentage of targets disabled and the percentage of friendly losses. However, the direction of this correlation is not as obvious. It may be that performing the mission more quickly increases the probability of success because of the surprise factor, and because there would be less opportunity to lose assets. However, it may also be that when the mission is completed in a shorter time, less of the targets are successfully found and engaged, decreasing the probability of success. Taking these correlations into account, it is expected that architectures which perform well in probability of success will correspond to a high rate of target disablement and low combat attrition. Since the two metrics time and probability can be calculated without an engagement model, and combat attrition and percentage of targets disabled would most likely require an engagement model, time to complete mission and probability of success will be used in RAAM in the early phases of evaluation. The other two metrics will be considered later using a simplified engagement model on a selection of the downselected architectures. Ease of maintainability and complexity, and acquisition costs will be considered qualitatively in RAAM, but complexity will later be calculated quantitatively for the final architectures. System acquisition cost will be calculated quantitatively using RAAM.

5.3 Gap Analysis

Once the metrics were mapped to the requirements for SEAD, gap analysis was performed. For each metric, estimations on the gap size were developed based on information contained in [21]. Criticality estimates were assigned notionally to the

requirements, and projected into the metrics space using the standard QFD approach. Each capability requirement was assigned an expected criticality on a one to five scale, and then given a minimum and maximum criticality as well. The gap sizes were then calculated on a metric basis, in order to enable a clear mapping between modeling results and gap closure. In order to do this, notional thresholds were set for each technical metric. Then, current performance was estimated by giving a mode value and range for each metric. The gap size was then calculated as the percent difference between the threshold value and the current performance. In cases where the current performance was better than the threshold, the gap size was set to zero, rather than having a negative gap size. This was done at the minimum value, the maximum value, and the mode value, giving the range of gap sizes. The criticalities was then mapped into the gap size space by projecting them through the mappings between the capabilities and metrics shown in Figure 70. It would have been equally possible to map the size of the gaps into the requirement space, and that may have been a more desirable approach for some problems. It should be noted that the cost-related metrics were not given an estimation of gap size. This is because it is assumed that decision-makers are most interested in the change in cost due to a proposed solution, and the effective benefit to cost ratio. Obviously once the costs are known solutions may be eliminated based on a cast threshold, but it is assumed that this will be determined later in the study and is not a focus of the initial gap analysis. A summary of the estimates of the current performance are shown in Figure 70. Mappings are done qualitatively at this point, were mappings between metrics and requirements and metrics and metrics are on a 1-3-5 scale, and the criticality is on linear 1-5 integer scale. This was used to implement the process described in 4.2.2, where is was assumed that decision-makers want to minimize the friendly losses and the time to complete mission, and maximize the probability of success and the percent of targets disabled. The results of the application of the process are shown in

											Probability of Success
				+5							% Targets Disabled
				-3	-5						Combat Attrition (% Friendly Losses)
				0	+3	+5					Time to Complete Mission
				0	0	0	0				Maintainability
				0	0	0	0	+1			System Acquisition Cost
				+1	+1	-1	-1	+3	0		Complexity
	Min Criticality	Average Criticality	Max Criticality	Probability of Success	% Targets Disabled	Combat Attrition (% Friendly Losses)	Time to Complete Mission	Maintainability	System Acquisition Cost	Complexity	
Max				0.5	70	2	280	n/a	n/a	n/a	
Average Performance				0.4	60	0.4	240	1.5	n/a	1	
Min				0.2	40	0	200	n/a	n/a	n/a	
Threshold				0.9	90	1	220	n/a	n/a	n/a	
Effectively disable air defenses	3	5	5	5	5						
Timely disablement of air defenses	1	2	4				5				
Reduce friendly losses	2	3	5			-5					
Reduce mission costs	3	4	5					3	1	5	

Figure 70: ROSETTA implementation of SEAD for Gap Analysis

Figure 71. The probability of success and the percentage of targets disabled were the top gaps, with time to complete mission being next in line followed by the combat attrition. However, because of the uncertainty, it is possible that these rankings would vary.

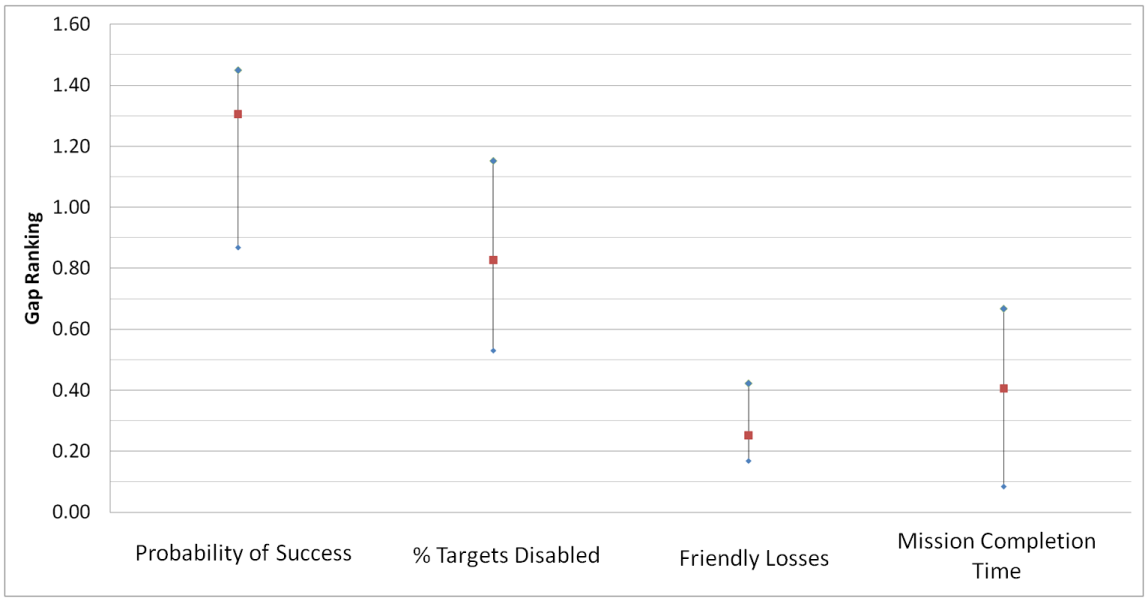


Figure 71: Gap Analysis Results for SEAD Mission

5.4 The ARCHITECT Environment

In order to aid a user in the execution of the remaining steps in the ARCHITECT methodology, an interactive environment was created using the JMP® statistical discovery and visualization software. Implementation in JMP® has the added benefit of providing a framework for keeping track of all assumptions, inputs, outputs and models used in the CBA processing. Automation is utilized wherever possible to decrease the total time required to perform the CBA. The ARCHITECT Environment consists of four interactive tabs, each corresponding to a workshop session that would need to be held with either SMEs or decision makers during the process of conducting the CBA. Each tab corresponds to a workshop session, and is dependent on the information collected in the tab preceding it as well as any models that were executed during the interim time. It is also assumed that before entering the initial workshop, the upfront legwork detailed previously such as the identification of candidate missions and metrics of interest and gap analysis has already been performed. The roadmap of the environment was previously shown in Figure 64.

5.5 Identifying Alternatives

5.5.1 Operational/Process Variations

The first step in identifying alternatives is to define the operational/process variations. This requires use of the baseline OV-5b, such as the one previously presented in Figure 68, in order to identify dependencies between tasks. Tasks can be related in several ways. A task can be identified as 'must precede' or 'must secede' another task. Tasks can also be identified within the hierarchy as subtasks of other tasks. This occurs when a high level task is comprised of a set of other subtasks that can be mapped to individual systems. Tasks can also belong to a main sequence of events or to a bypass sequence that is executed in the event that specific conditions are met or not met. By adhering to these rules, alternate task sequences can be generated to represent variations in the overall manner in which the capability is accomplished. The ARCHITECT Environment supports this through the use of a graphical user interface (GUI), which is shown in Figure 72. The task mappings from the baseline are automatically loaded and can be edited in the GUI. Additional tasks can be added, and the mappings can be specified. In the end, all process ordering information should be captured, and tasks should be broken down to a sufficient level of detail that they can be mapped to individual systems. Figure 73 provides examples of how this is accomplished in a manner that supports automatic input to the ARCHITECT Environment.

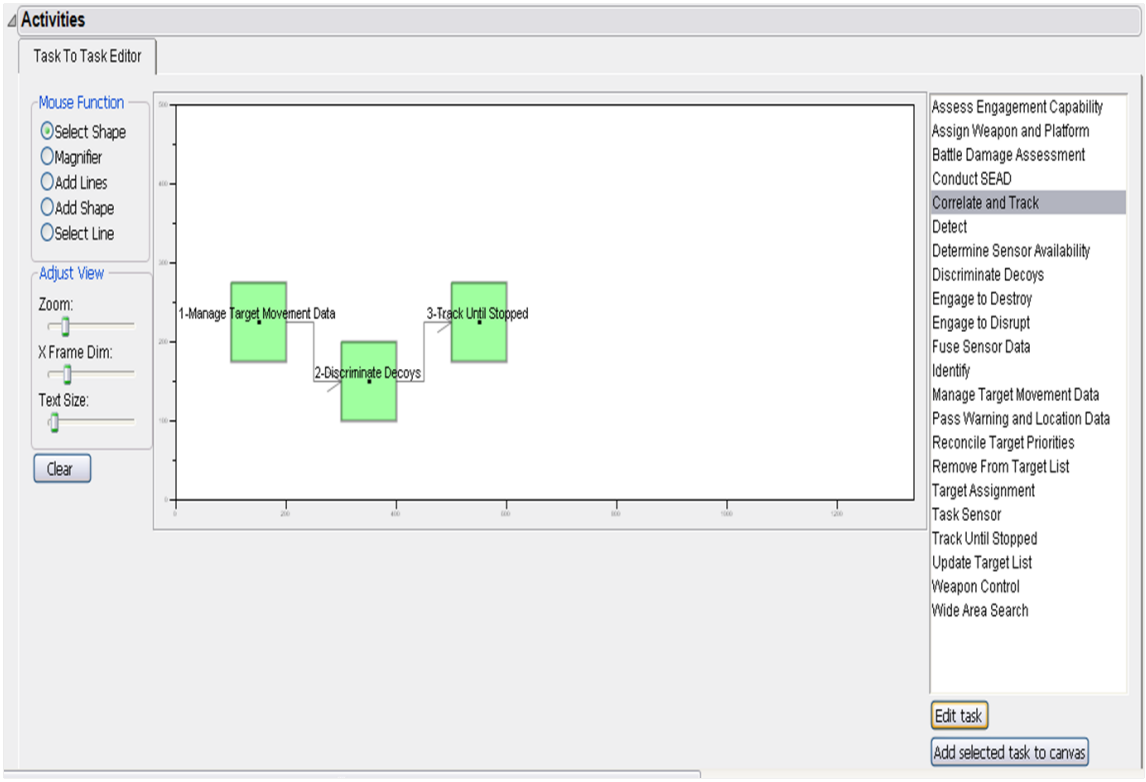


Figure 72: Operational Activity Alternative GUI

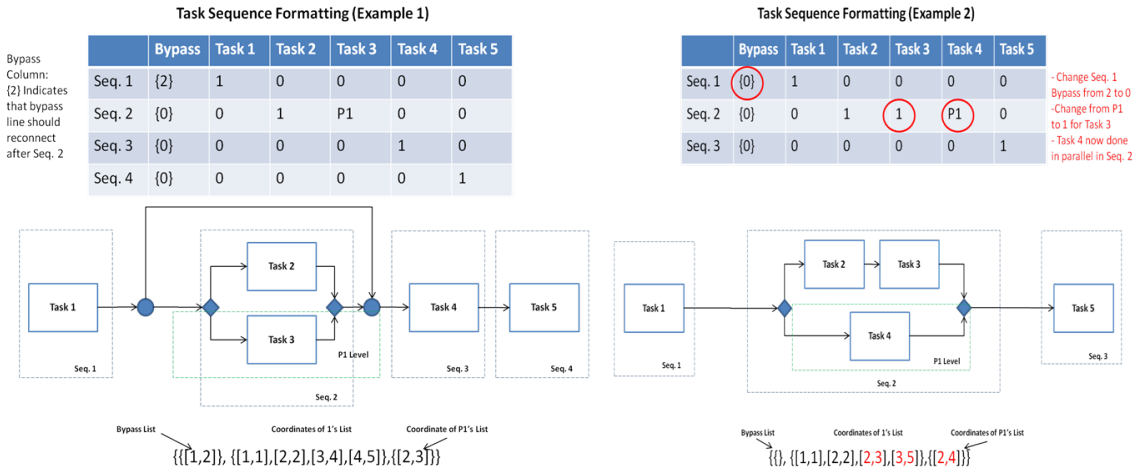
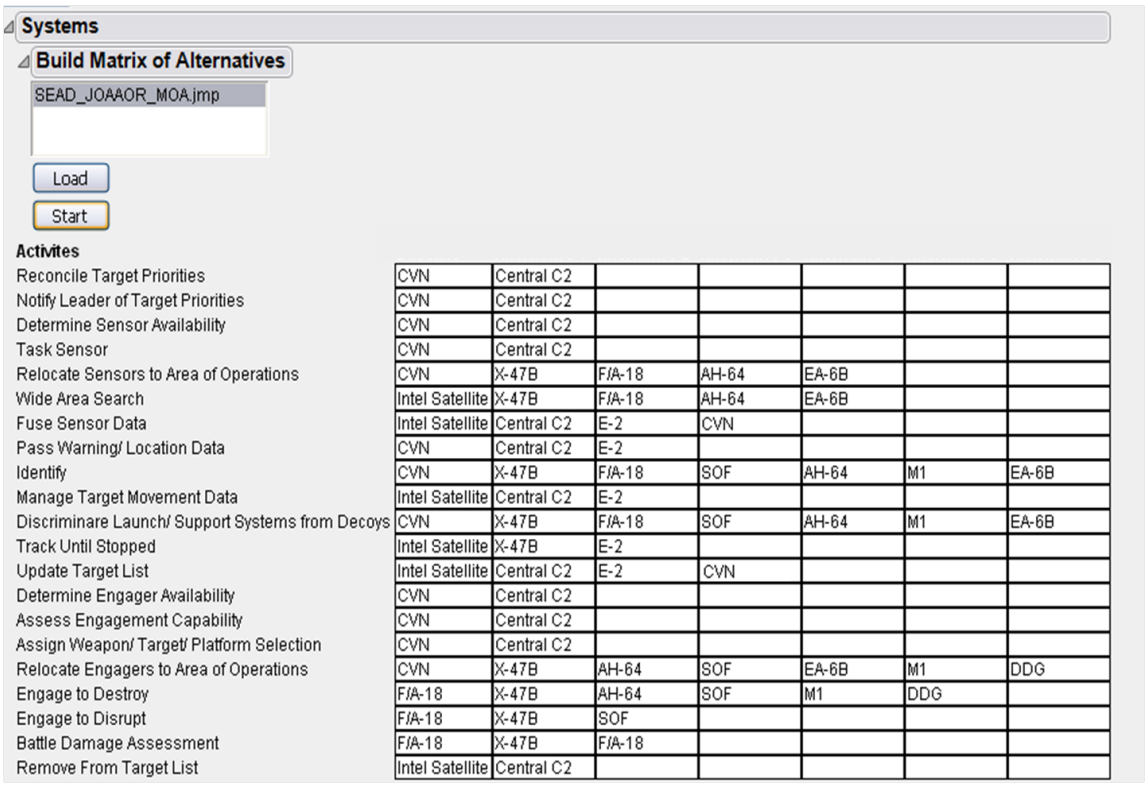


Figure 73: Examples of Conversion of OV-5 to Computer Readable Format. Reproduced from [59]

5.5.2 System Level Alternatives

Once the task mappings are created, the system/technology alternatives for each task are defined. This step is performed using a matrix of alternatives that is automatically generated by the ARCHITECT Environment and can be seen in Figure 74. Each task is given a row in the matrix, and the baseline systems are identified for each task from the baseline architecture views. The user then defines a set of alternative architectures made up of different combinations of new and existing systems that will perform the various tasks. It is possible that the same system will map across multiple tasks. While the user is free to specify as many alternatives per task as desired, users must be cautioned that specifying more than five or six alternatives per task will result in a very large number of possible system portfolios since the alternative space is combinatorial in nature, Once this is completed, the user then needs to specify the performance estimates for each system and system-task pair listed in the matrix of alternatives.



Activites							
Reconcile Target Priorities	CVN	Central C2					
Notify Leader of Target Priorities	CVN	Central C2					
Determine Sensor Availability	CVN	Central C2					
Task Sensor	CVN	Central C2					
Relocate Sensors to Area of Operations	CVN	X-47B	F/A-18	AH-64	EA-6B		
Wide Area Search	Intel Satellite	X-47B	F/A-18	AH-64	EA-6B		
Fuse Sensor Data	Intel Satellite	Central C2	E-2	CVN			
Pass Warning/ Location Data	CVN	Central C2	E-2				
Identify	CVN	X-47B	F/A-18	SOF	AH-64	M1	EA-6B
Manage Target Movement Data	Intel Satellite	Central C2	E-2				
Discriminare Launch/ Support Systems from Decoys	CVN	X-47B	F/A-18	SOF	AH-64	M1	EA-6B
Track Until Stopped	Intel Satellite	X-47B	E-2				
Update Target List	Intel Satellite	Central C2	E-2	CVN			
Determine Engager Availability	CVN	Central C2					
Assess Engagement Capability	CVN	Central C2					
Assign Weapon/ Target/ Platform Selection	CVN	Central C2					
Relocate Engagers to Area of Operations	CVN	X-47B	AH-64	SOF	EA-6B	M1	DDG
Engage to Destroy	F/A-18	X-47B	AH-64	SOF	M1	DDG	
Engage to Disrupt	F/A-18	X-47B	SOF				
Battle Damage Assessment	F/A-18	X-47B	F/A-18				
Remove From Target List	Intel Satellite	Central C2					

Figure 74: Matrix of Alternatives to Define SEAD System Alternative Space

5.5.3 Organizational Considerations

Next, the organizational alternative space is defined by designating the organization which owns and operates each system. For baseline systems, this will be known automatically from the baseline architecture, but will need to be user-supplied for systems not included in the baseline. This is done for the SEAD mission within the ARCHITECT Environment using the GUI shown in Figure 75.

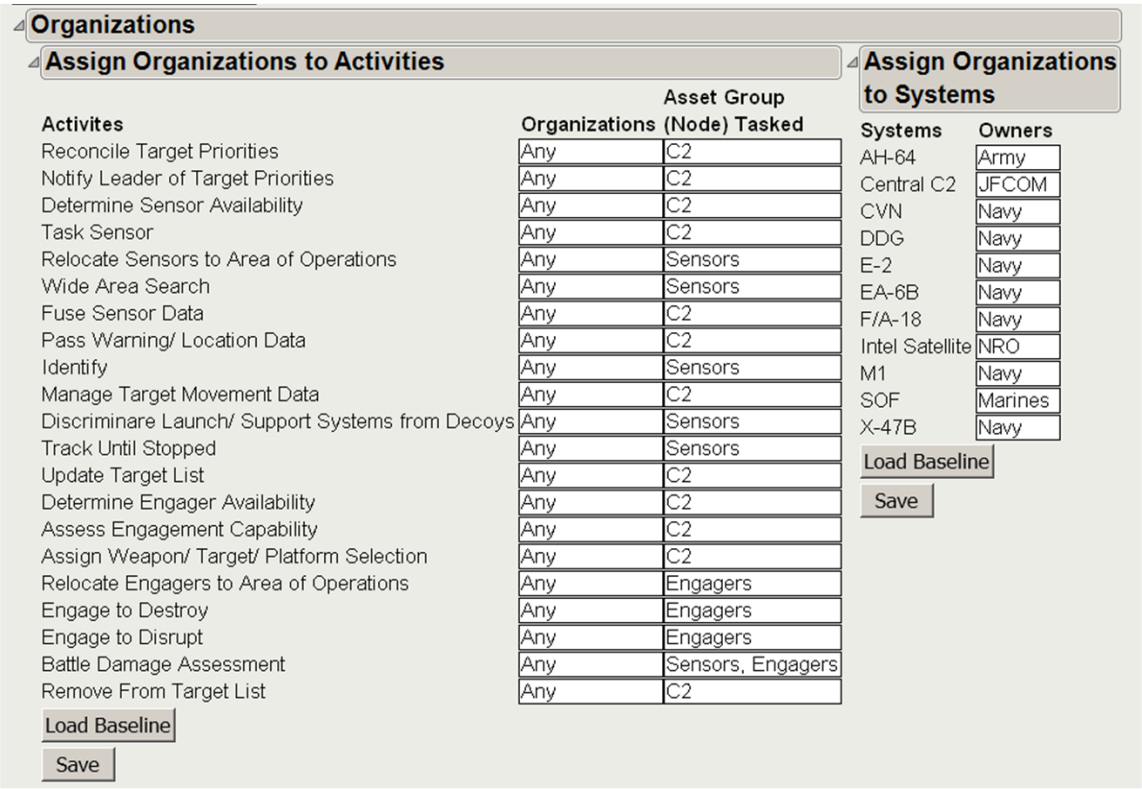


Figure 75: Organizational Constraints for SEAD Example

5.6 Alternative Evaluation

The first step of the Alternative Evaluation process is to collect estimates of systems performing tasks to use as an input to RAAM. Four metrics were identified as being compatible with the RAAM framework, and estimates were made for every system and every system-task pair. These are provided in Appendix B. Next, RAAM was run to obtain first-order estimates of these four metrics, which included acquisition cost, risk, average time to execute the SEAD kill chain, and qualitative maintainability. Maintainability and complexity represent qualitative metrics, and time to complete and probability of success are representative of quantitative metrics. A summing aggregation function was used for time, a product function was used for probability of success, and a minimum function was used for maintainability. For complexity, a more complicated aggregation was used that calculated the complexity for each

segment of the mission. Since the total alternative space had over 700,000,000 feasible architectures, it was decided to first group the alternatives by their system portfolios and eliminate portfolios with overall poor performance. In order to do this, a dynamic visualization environment was created to help understand the results of the data. The visualization environment presents the data using multiple types of charts and displays, and is interactive, allowing the user to interact with the data. Furthermore, the displays are linked together, so that highlighting a particular data point or group of points in one view will highlight the same point in other views, thus allowing a more in-depth exploration and understanding of the data.

5.6.1 Results from RAAM Portfolio Analysis

Upon implementing RAAM, 1,266 portfolios were found to be feasible and were evaluated against the four initial metrics. This data was then imported into the JMP® statistical software package for analysis. A visualization environment was created to help decision-makers determine which portfolios to eliminate from further consideration. Several different visualizations for analysis were created to allow decision-makers and engineers to work together, including a scatterplot matrix (shown in Figure 76), distributions of the results for every metric and an OEC which used an equal weighting scheme on all metrics (shown in Figure 77), distributions for each system on how often each was included and excluded from portfolios, a data filter allowing filtering by any input or output variable, and a prediction profiler and Pareto plots to enable sensitivity analysis (shown in Figure 78). The baseline performance is also included on the scatterplots, and is represented by the red star. There were several steps taken in the elimination of portfolios. It should be noted that decision-makers have the ability to change the weightings on the OEC on-the-fly.

The scatterplot matrix is a triangular matrix made up of scatterplots of every response against every other response. In each of the scatterplots, every portfolio

is represented by a point, showing the average performance of that portfolio on one metric against the average performance of that portfolio on another metric. All 1,266 portfolios are represented in each of the scatterplots. Highlighting any one portfolio in one plot will cause that same portfolio to be highlighted in all other plot, thus giving the ability to visualize all of the dimensions of the problem simultaneously. Any changes made to a portfolio in one scatterplot (such as changing the point color or the marker shape) will be reflected in that portfolio across all other scatterplots as well. All of the information about each portfolio (i.e., which systems are included, the performance estimates for those systems that were input to the model, and the performance estimates for that portfolio across all metrics) is stored with each point, and can be easily pulled up if needed.

Once the visualization environment was created, filtering was applied to the results, to eliminate all portfolios with low average performance against the metrics. Within the environment, filters can be applied to one metric at a time, or to multiple metrics at the same time (where the filters are joined by an AND statement). In addition, filters can be applied to systems (i.e. filter all portfolios that do or do not include system x). This allows decision-makers to examine the problem from either a top-down or a bottom-up perspective. As an example of metric filtering, the application of two constraints, one for probability of success greater than 0.4 and one for time to completion less than 220 minutes, reduced the space to only 166 portfolios, which are shown in Figure 79. These remaining portfolios are those that had both a completion time of less than 220 minutes, and a probability of success of greater than 0.4. Looking at the system distributions for the Central C2 and the CVN after this initial downselection (as shown in Figure 80 shows some interesting results. First, it should be noted that the light green region in the figure represents the total distribution of how many of the total 1,266 portfolios include (1) and don't include (0) each of the systems. The shaded dark green represents the same distribution for only the

remaining 166 portfolios. It is interesting to see that almost all of the remaining 166 portfolios include the Central C2 and do not include the CVN. This suggests that for this mission, with these performance thresholds, the CVN is not an appropriate system, and that the Central C2 is a necessary choice. This can also be seen in two other ways as well. First, a sensitivity analysis is run which, for each metric, shows the relative impact of including each system on the variability of the value of that metric. The results are displayed in a tornado chart, which ranks the systems from the most impactful to the least impactful, and uses a bar chart to show both the magnitude and direction of each system's impact of the variability. In the sensitivity analysis, the tornado chart for time to completion (shown in Figure 81), the CVN has the most impact on the result, and works to increase the total time, which is an undesirable result. The Central C2 has the second most impact, and has a positive result, working to decrease the total time to complete the mission. Since the CVN and the Central C2 are the only two options for many of the tasks, it then follows that the Central C2 would be the necessary choice.

With this in mind, the filters can be reset to again consider all 1,266 portfolios. Then, applying filters to exclude the CVN and force the inclusion of the Central C2 shows the impact of this decision on the results. Figure 82 shows the distributions of the responses, where the light green represents all 1,266 of the portfolios and the shaded dark green represents those cases where the Central C2 is included and the CVN is not. Note that the application of these filters will include architectures that were not included when applying the metrics-based filters, as not all of the cases that include Central C2 and exclude the CVN necessarily meet the thresholds used previously. There are a total of 422 portfolios which meet these filters. There is a clear shift in the mean on the average time to completion (towards a shorter completion time), as well as on the ease of maintainability (toward more maintainable architectures) when these two selections are made. This makes sense, as these choices impact

the completion time as discussed previously, and the CVN is difficult to maintain. Further, re-applying the filters previously used on the metrics along with current filter for the exclusion of CVN and inclusion of Central C2, results in 144 remaining portfolios. This is one possible approach decision-makers might take to downselecting the portfolios to make the alternative space more manageable. Decision-makers might, at this point, choose to carry these 144 portfolios forward to the next level of the hierarchical evaluation and downselection. However, in this downselection approach, priority was given to the time to complete the mission. Remembering that the probability of success was likely to be a more significant gap in the gap analysis, decision-makers might decide to explore an alternative downselection process using this data, which gives higher priority to the probability of success.

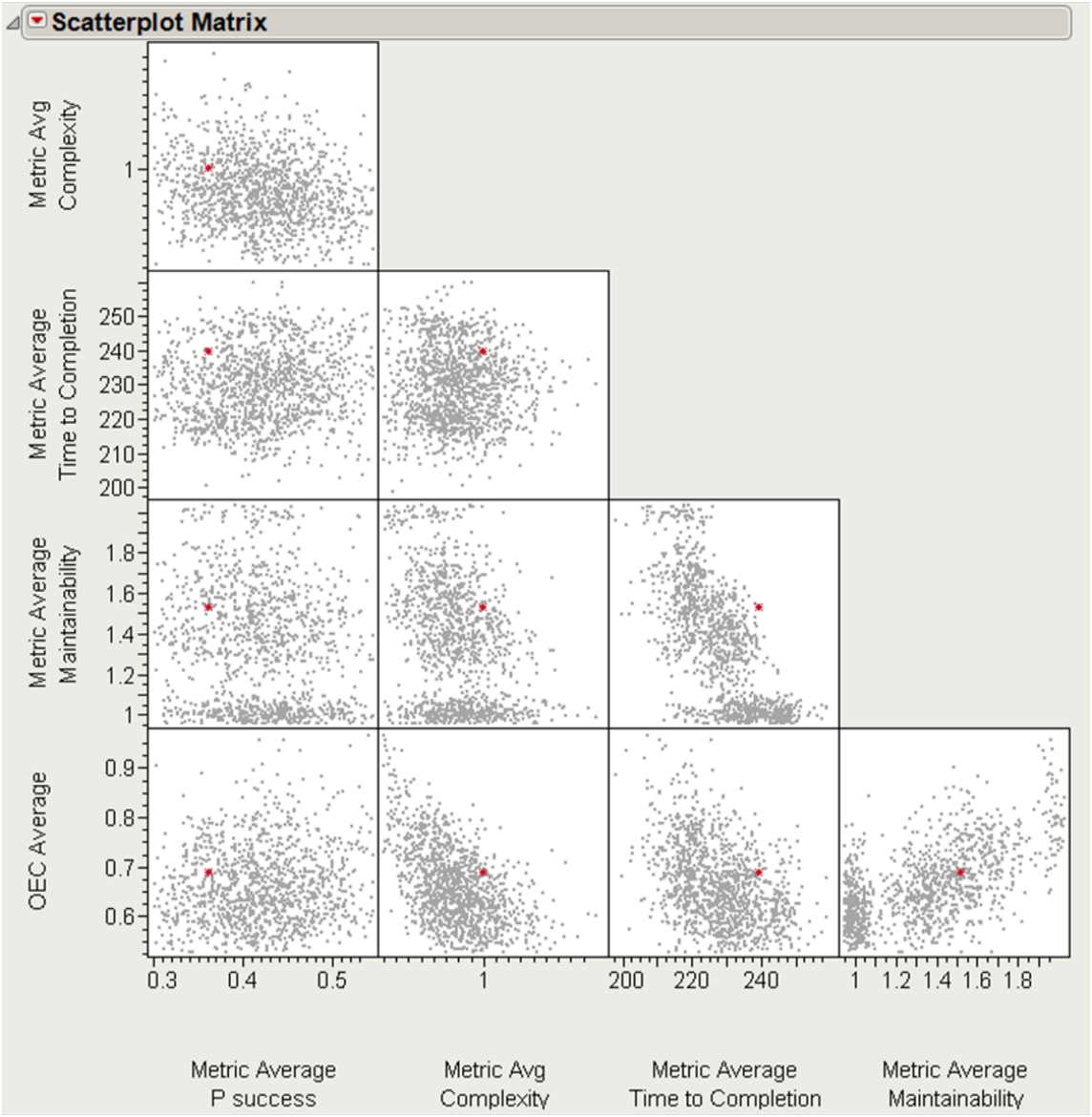


Figure 76: Scatterplot Matrix for SEAD Portfolio Analysis for SEAD Portfolio Analysis

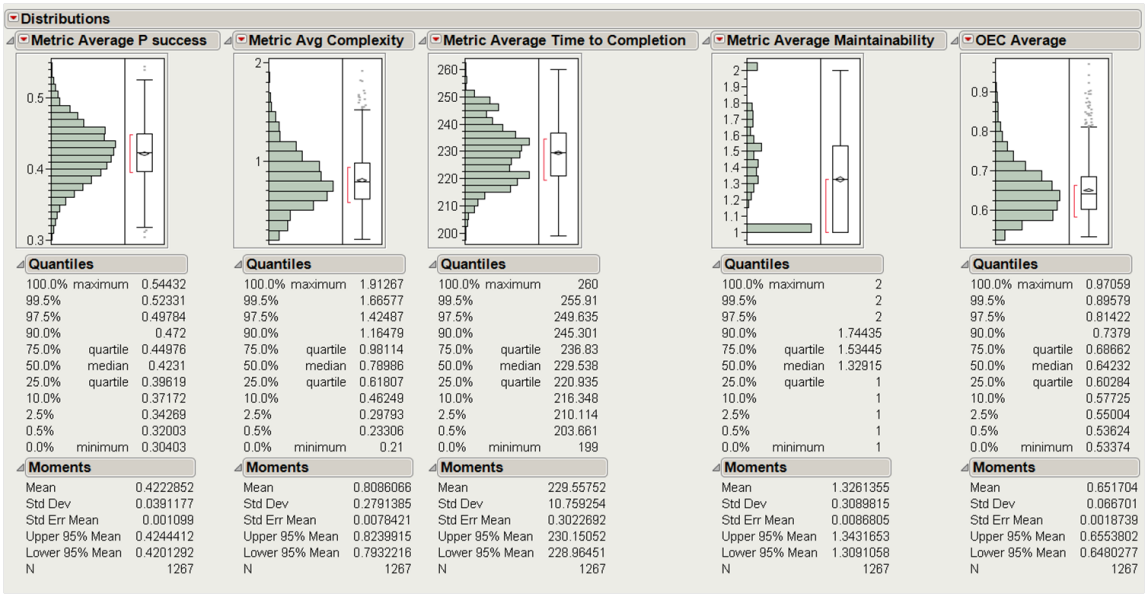


Figure 77: Distributions of Outputs for SEAD Portfolio Analysis

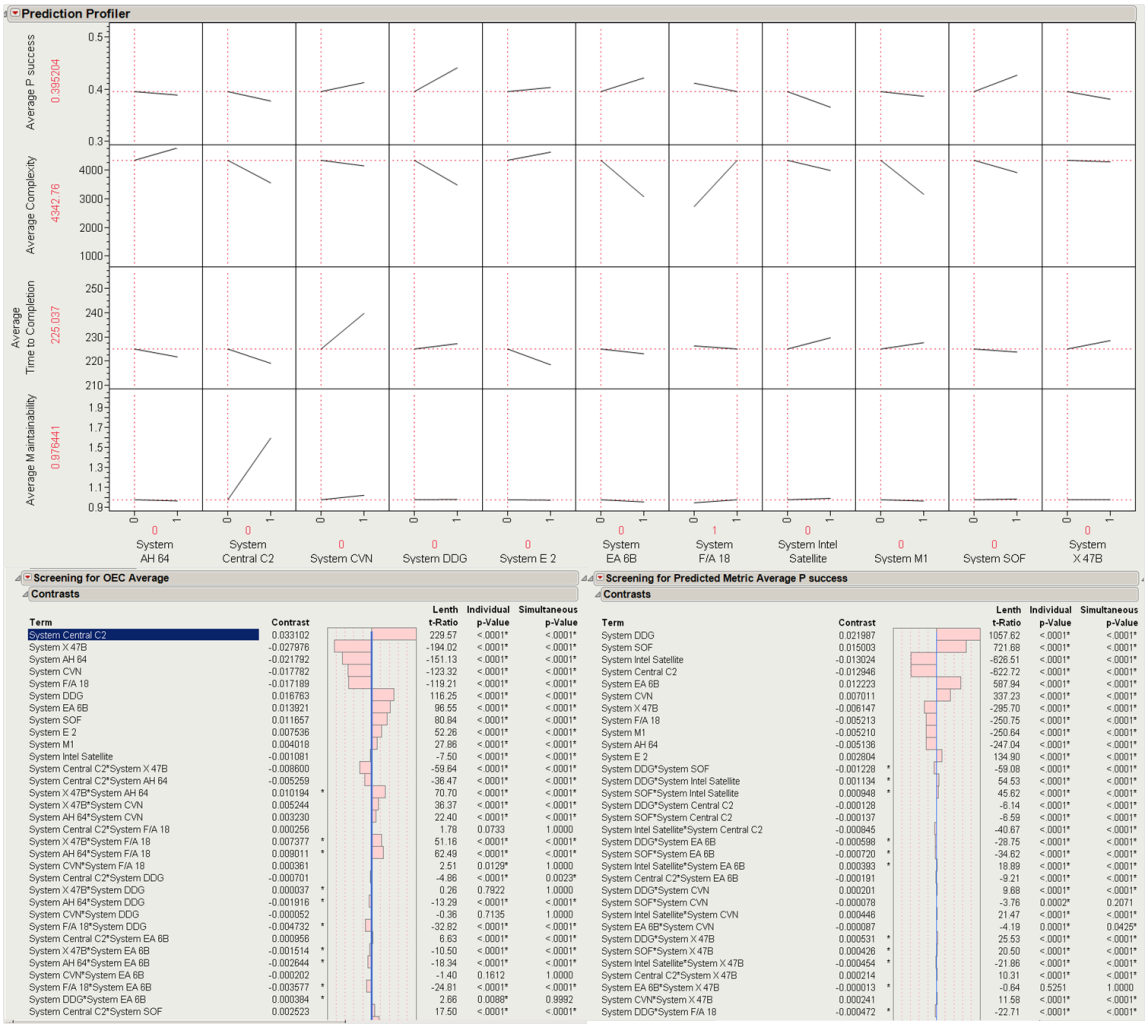


Figure 78: Prediction Profiler and Two of Five Pareto Plots for SEAD Portfolio Analysis

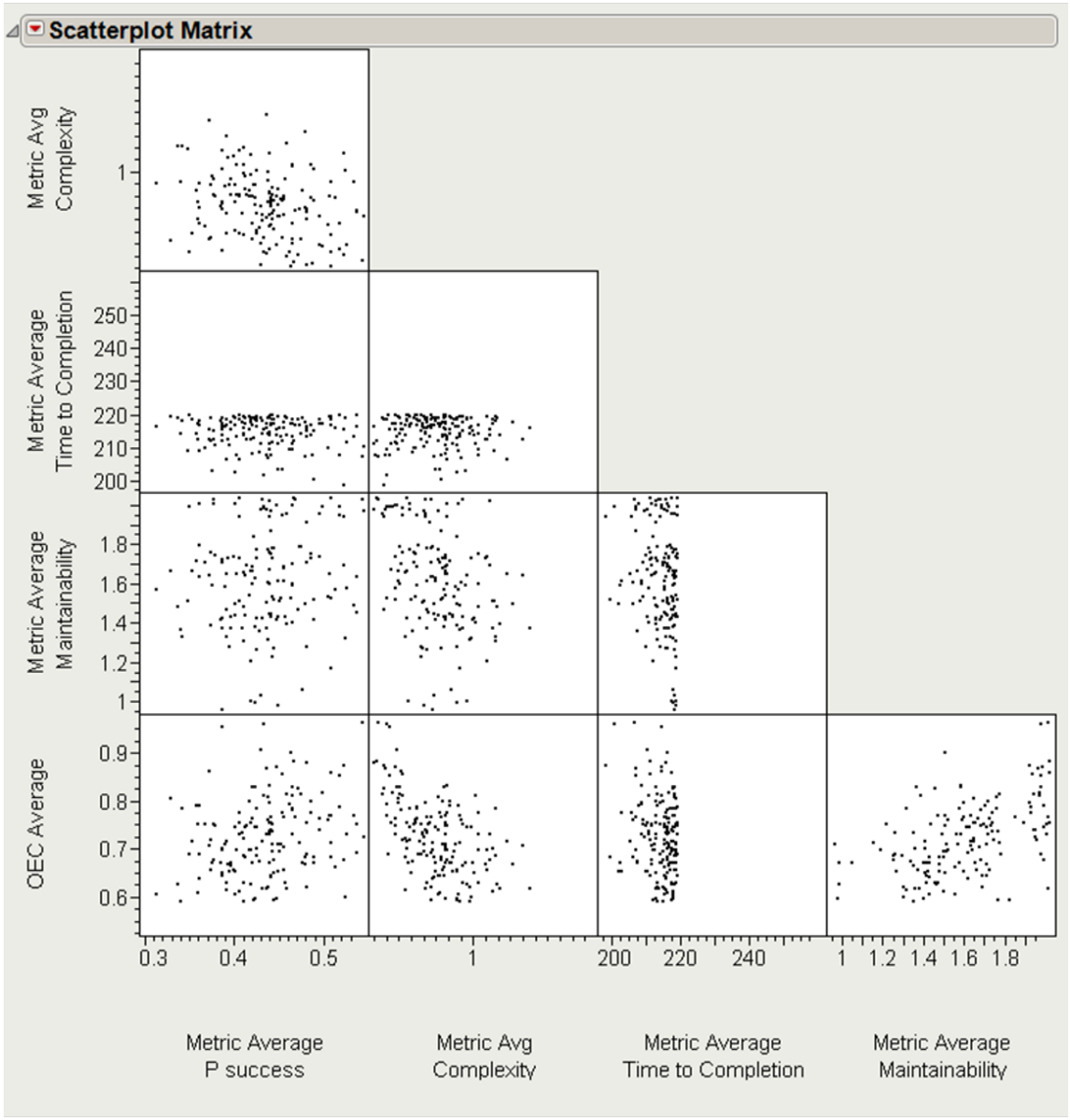


Figure 79: Remaining Portfolios after Initial Filtering

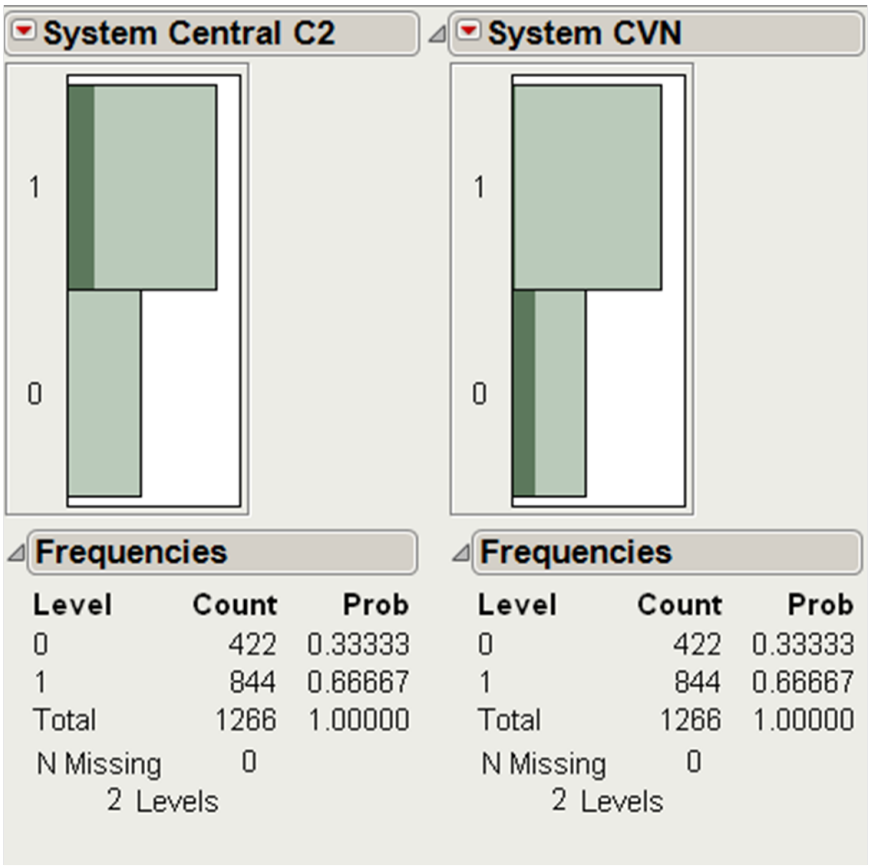


Figure 80: CVN and Central C2 Distributions after Initial Filtering

Screening for Predicted Metric Average Time to Completion

Contrasts

Term	Contrast		Lenth t-Ratio	Individual p-Value	Simultaneous p-Value
System CVN	7.61005		1992.67	<.0001*	<.0001*
System Central C2	-4.10381		-1074.6	<.0001*	<.0001*
System E 2	-3.88249		-1016.6	<.0001*	<.0001*
System Intel Satellite	2.10793		551.95	<.0001*	<.0001*
System X 47B	1.98739		520.39	<.0001*	<.0001*
System AH 64	-1.54822		-405.40	<.0001*	<.0001*
System M1	1.11915		293.05	<.0001*	<.0001*
System DDG	0.97876		256.29	<.0001*	<.0001*
System EA 6B	-0.91603		-239.86	<.0001*	<.0001*
System F/A 18	-0.65518		-171.56	<.0001*	<.0001*
System SOF	-0.64597		-169.15	<.0001*	<.0001*
System CVN*System E 2	-0.84099		-220.21	<.0001*	<.0001*
System Central C2*System E 2	0.23384		61.23	<.0001*	<.0001*
System CVN*System Intel Satellite	0.25757		67.44	<.0001*	<.0001*
System Central C2*System Intel Satellite	-0.02699		-7.07	<.0001*	<.0001*
System E 2*System Intel Satellite	-0.09508	*	-24.90	<.0001*	<.0001*
System CVN*System X 47B	0.33300		87.20	<.0001*	<.0001*
System Central C2*System X 47B	-0.14303		-37.45	<.0001*	<.0001*
System E 2*System X 47B	-0.10509	*	-27.52	<.0001*	<.0001*
System Intel Satellite*System X 47B	0.07259	*	19.01	<.0001*	<.0001*
System CVN*System AH 64	-0.42337		-110.86	<.0001*	<.0001*
System Central C2*System AH 64	0.05978		15.65	<.0001*	<.0001*
System E 2*System AH 64	0.12782	*	33.47	<.0001*	<.0001*
System Intel Satellite*System AH 64	-0.04321	*	-11.31	<.0001*	<.0001*
System X 47B*System AH 64	-0.05755	*	-15.07	<.0001*	<.0001*
System CVN*System M1	0.21765		56.99	<.0001*	<.0001*
System Central C2*System M1	-0.04482		-11.74	<.0001*	<.0001*
System E 2*System M1	-0.05625	*	-14.73	<.0001*	<.0001*
System Intel Satellite*System M1	0.03274	*	8.57	<.0001*	<.0001*
System X 47B*System M1	0.05656	*	14.81	<.0001*	<.0001*
System AH 64*System M1	0.00469	*	1.23	0.2204	1.0000
System CVN*System DDG	0.51154		133.95	<.0001*	<.0001*

Figure 81: Pareto Plot for Time to Complete Mission

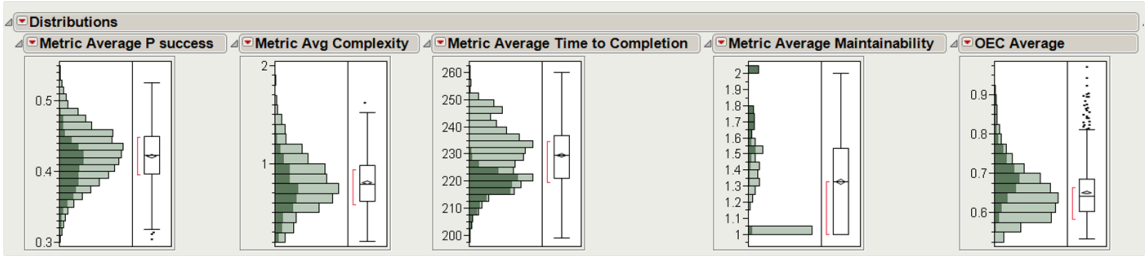


Figure 82: Result Distributions with CVN Excluded and Central C2 Included

Giving priority to probability of success shows an interesting trade. Although the CVN negatively impacts the time to mission completion, it favorably impacts the probability of success. In fact, selecting only the portfolios that make up the 90% quantile for probability of success (greater than a .47 probability of success) and then examining the resulting system distributions reveals that the ten percent of cases with the highest probability of success all include the CVN and the DDG, and exclude the Intel Satellite.

Using this information, filtering was then applied to consider only the 200 portfolios which include the CVN and DDG and exclude the Intel Satellite. The mean of the probability of success distribution is shifted up, but that all of those cases with a low time mission completion time have been eliminated. Thus, there is a clear trade between probability of success and time to completion. Furthermore, the ease of maintainability is expected to be low, which follows from the inclusion of the CVN in these 200 portfolios. This is all shown in the distributions on the metrics in Figure 83, where again the light green represents all 1,266 cases and the dark green represents the 200 portfolios remaining.

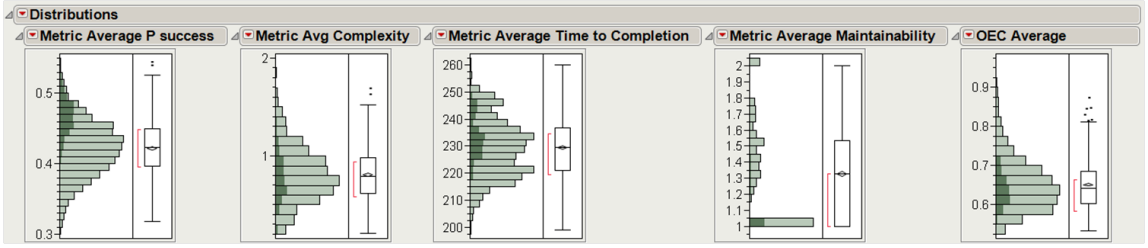


Figure 83: Result Distributions with CVN and DDG Included and Intel Satellite Excluded

The analysis enabled by the combination of RAAM and the dynamic visualization environment allows decision-makers to visualize these trades and determine which portfolios to carry forward in the analysis process, and which to eliminate. It can even result in the elimination of certain systems from consideration all together, which will eliminate large numbers of portfolios. Since the alternative space is subject to uncertainty and is very large, the goal is not necessarily to try and find an optimum solution, but rather at each stage of the downselection bias the included solution set to those solutions most likely to close the originally stated gaps, thus allowing decision-makers to focus on the regions of the solution space most likely to contain solutions that will meet their needs.

In this case, it will be assumed that the decision-makers decided, based on the analysis above, to tighten the constraint on probability of success and take only those portfolios in the top ten percent, and to relax the constraint on the time to complete the mission to 220 minutes. These constraints were applied as filters, resulting in a total of 16 remaining portfolios, which were deemed to have the best chance of meeting decision-maker needs. The remaining 16 portfolios' performance is shown in Figure 84 and the portfolios themselves are enumerated in Figure 85. It is interesting to note that all of the final portfolios included the EA-6B, and none included the F/A-18 or the Intel Satellite. Most included the E-2. Since this is notional data, this does not likely reflect the results that one would get for the real mission or systems, but is the result based on this example case. Since the goal of the CBA is to determine whether or not a materiel solution is needed, it is of interest to note that most of the remaining alternatives did not include the acquisition of the X-47B, a new materiel solution that is partway through the development process. However, since one of the remaining portfolios did include this system, it is still necessary to consider this materiel solution moving forward. The initial results do suggest that similar performance can be gained from non-materiel solutions which use existing assets in new ways, which gives an

initial indication that in this case study, DOT_LPF changes may be the preferred path. For this example implementation of ARCHITECT, these decisions represent the end of the first level of the hierarchical evaluation and downselection process, and the remaining portfolios will be taken to the next level of the analysis.

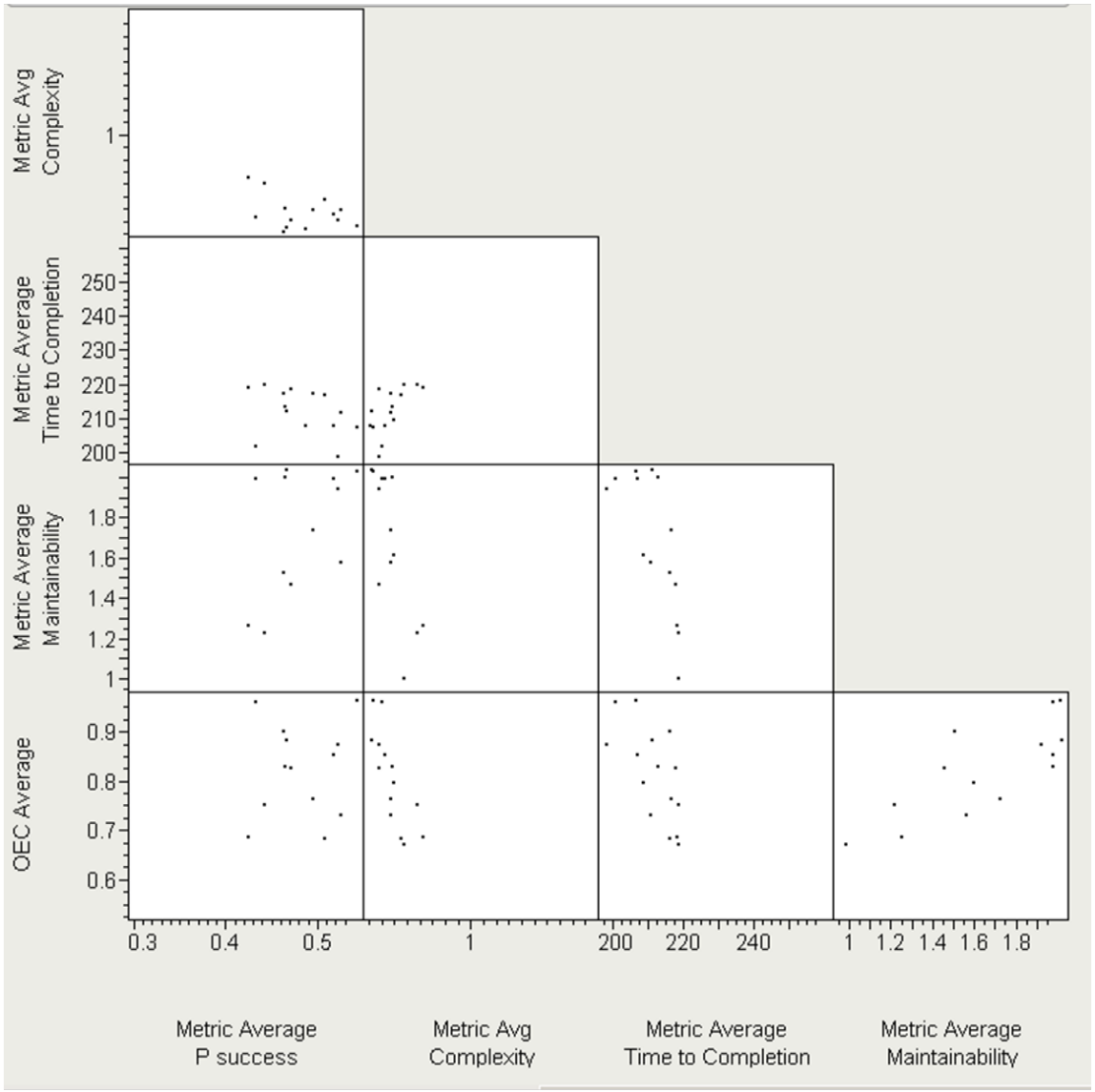


Figure 84: Performance of Final 16 SEAD Portfolios

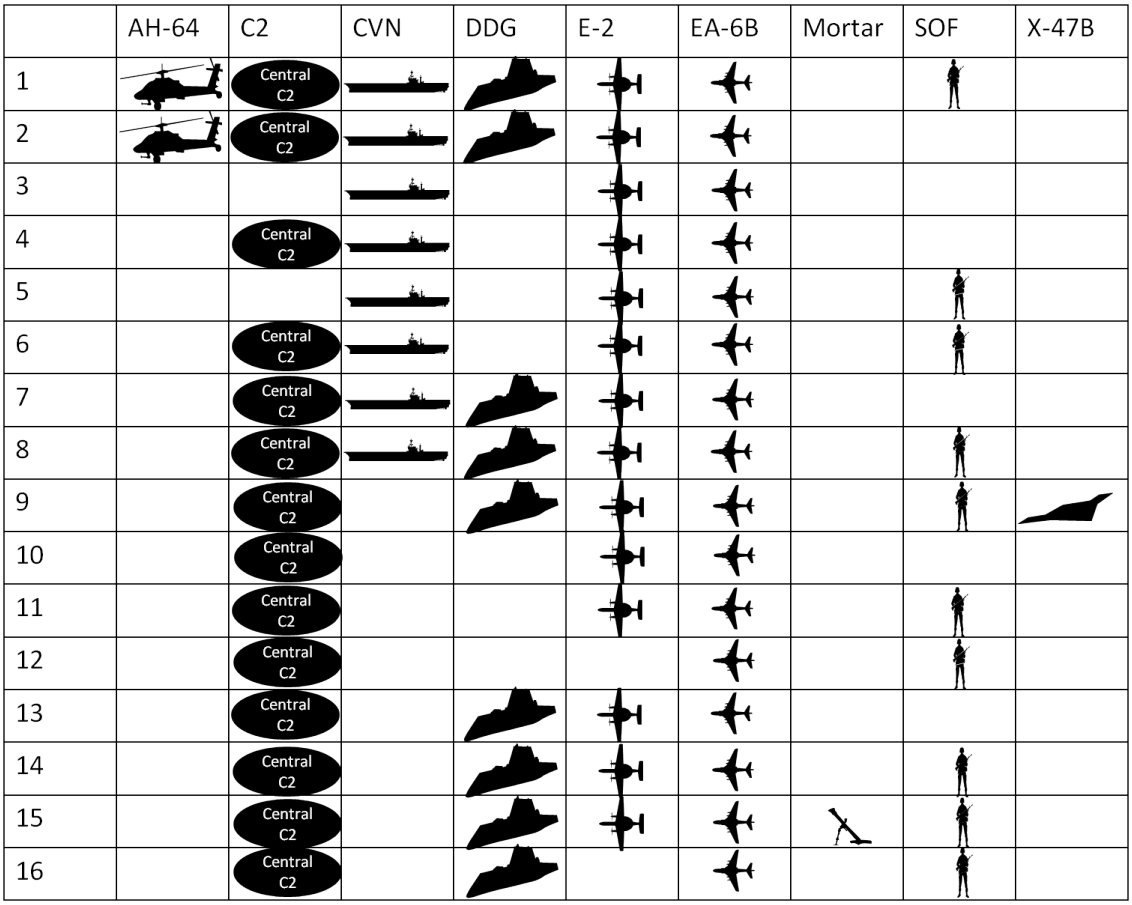


Figure 85: Composition of Final 16 SEAD Portfolios

It should be noted that this is one example of how this initial downselection could be done. Decision makers could have also eliminated systems based on additional information, such as cost or TRL, and then used the RAAM results to select portfolios from the remaining options. Decision-makers with a different set of priorities would likely have gone through the downselection process differently, as was demonstrated using the initial downselection approach presented here. Rather than simply use information on the average performance of portfolios, the decision-makers could have instead looked at the range of performance for the portfolios, and used this information to help with the downselection process. The process shown here is meant to demonstrate how the techniques used by the ARCHITECT method enable a more rigorous and transparent decision-making process which examines a broader range of alternatives in more quantitative analysis format.

5.6.2 Results from Second Round of RAAM Analysis

Once these 16 portfolios were selected, the next step was to re-run them through RAAM and obtain all of the possible operational use cases of these portfolios in conducting the SEAD mission. This means that for each portfolio, every possible system-task mapping thread through the mission was run independently, and the results were not averaged across the portfolio. This allowed operational use cases that did not perform to be ruled out, leaving only a subset of the remaining operational-system alternative set to be evaluated using other modeling techniques. The downselection process used in this example implementation of ARCHITECT is described below; however,like the previous downselection, the choices and results will be dependent on the decision-maker and what is of interest. In addition to the other four metrics, acquisition cost (or value based on acquisition price) of each portfolio was also included in the calculations so as to give another way to differentiate between alternatives and further refine the downselection. The full results of this second round

of runs are shown in Figure 86. The baseline is shown using a black star. All of the alternatives within each portfolio are grouped by color. As can be seen in the figure, no single portfolio stands out as being necessarily superior. However there are some initial observations that can be made from this figure. First, some portfolios have a much greater variability than others when deployed using different system-task mappings. Some portfolios have a greater number of possible system-task mapping variations. The portfolios are grouped into bands with respect to both cost and maintainability, suggesting that certain systems are driving these metrics. Furthermore, portfolios 1 and 2 seem to have multi-modal behavior with respect to the probability of success, suggesting that certain system-task pairings are driving up (or down) the probability of success. All of these observations can be explored further using the visual environment to attempt to determine the causes of the observed behaviors and give insight to the decision-makers.

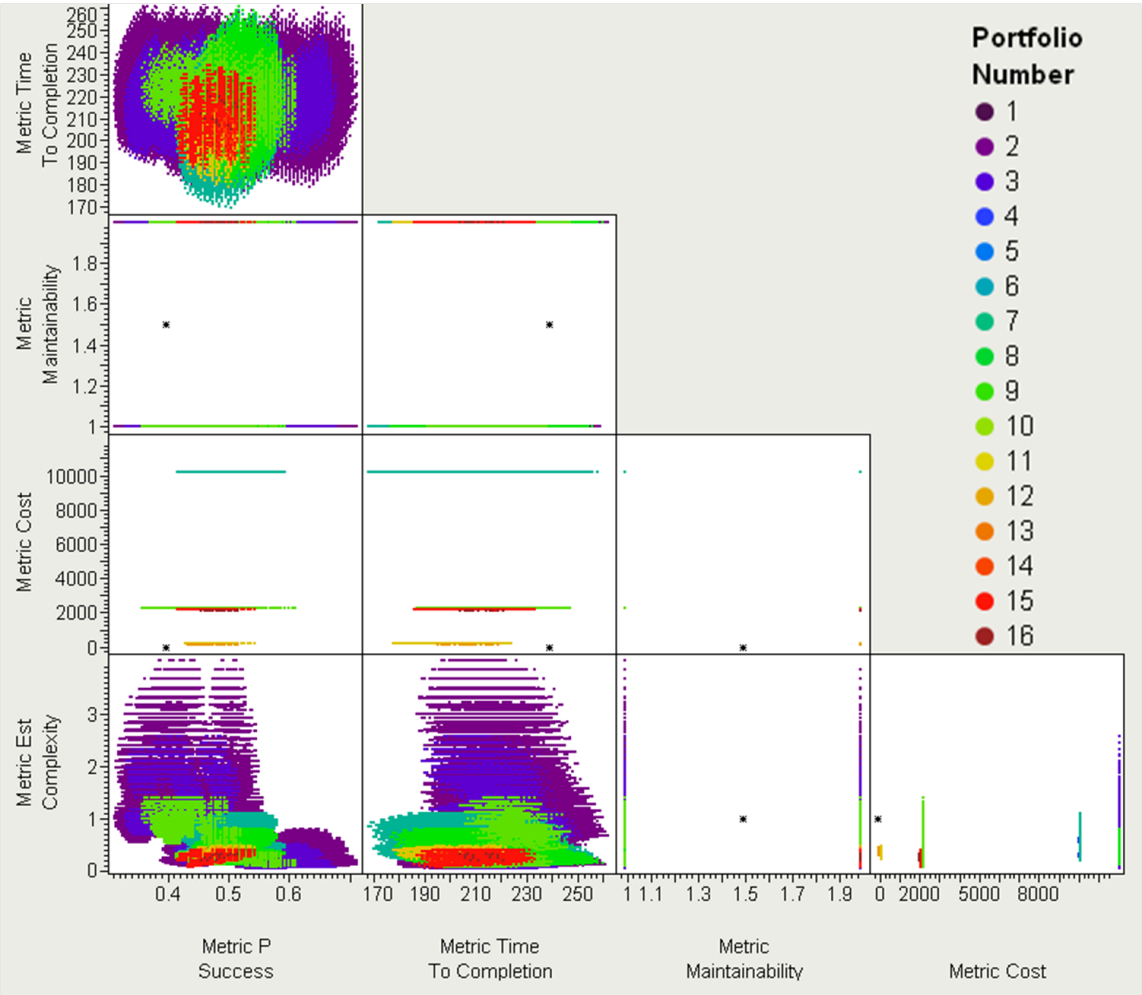


Figure 86: Results for the 16 Remaining Portfolios, Colored by Portfolio

The distributions on the results across all of the cases of the 16 portfolios are shown in Figure 87. However, as these results include all cases for all portfolios, they are dominated by the results of portfolios 1 and 2, which have the highest number of possible architectures. Thus, rather than look at the total distribution, it is of interest to look at the distributions on a portfolio by portfolio basis. These distributions are shown in Figures 88 through 91. The mean and standard deviation for all portfolios across all metrics are shown in Table 12. This information can also be displayed in bar chart form, as shown in Figures 92 through 95. Again, it is clear that no single portfolio stands out. This is expected as the portfolios selected to carry forward were those that overall met a given set of performance criteria. Thus, a deeper level of exploration is required to determine what operational cases (if any) to carry forward within each portfolio.

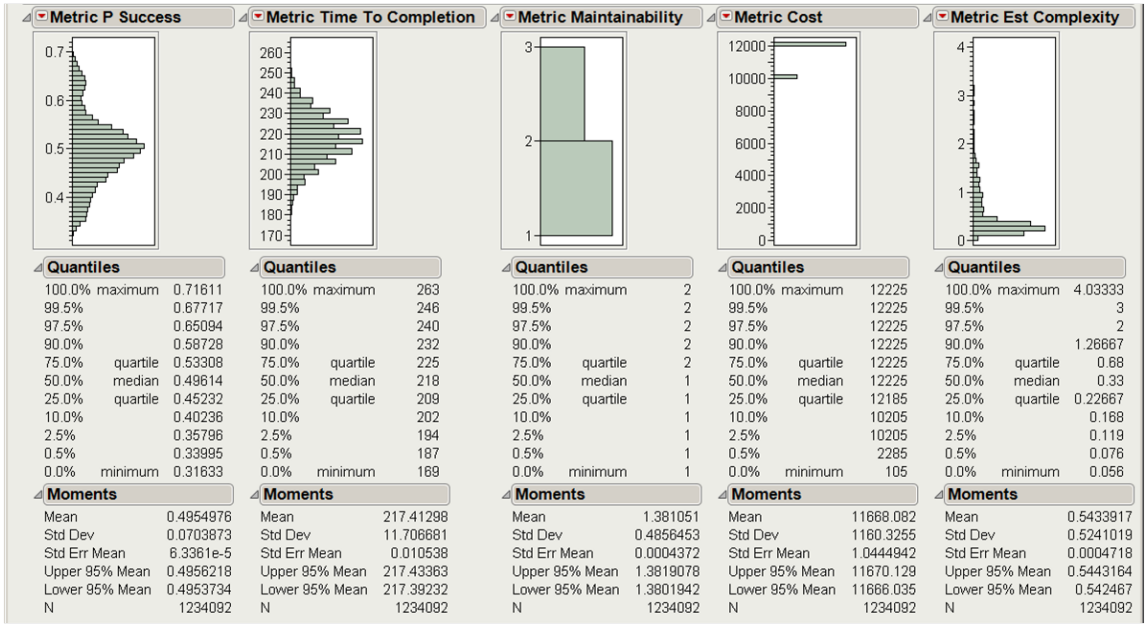


Figure 87: Distributions for the 16 Remaining Portfolios

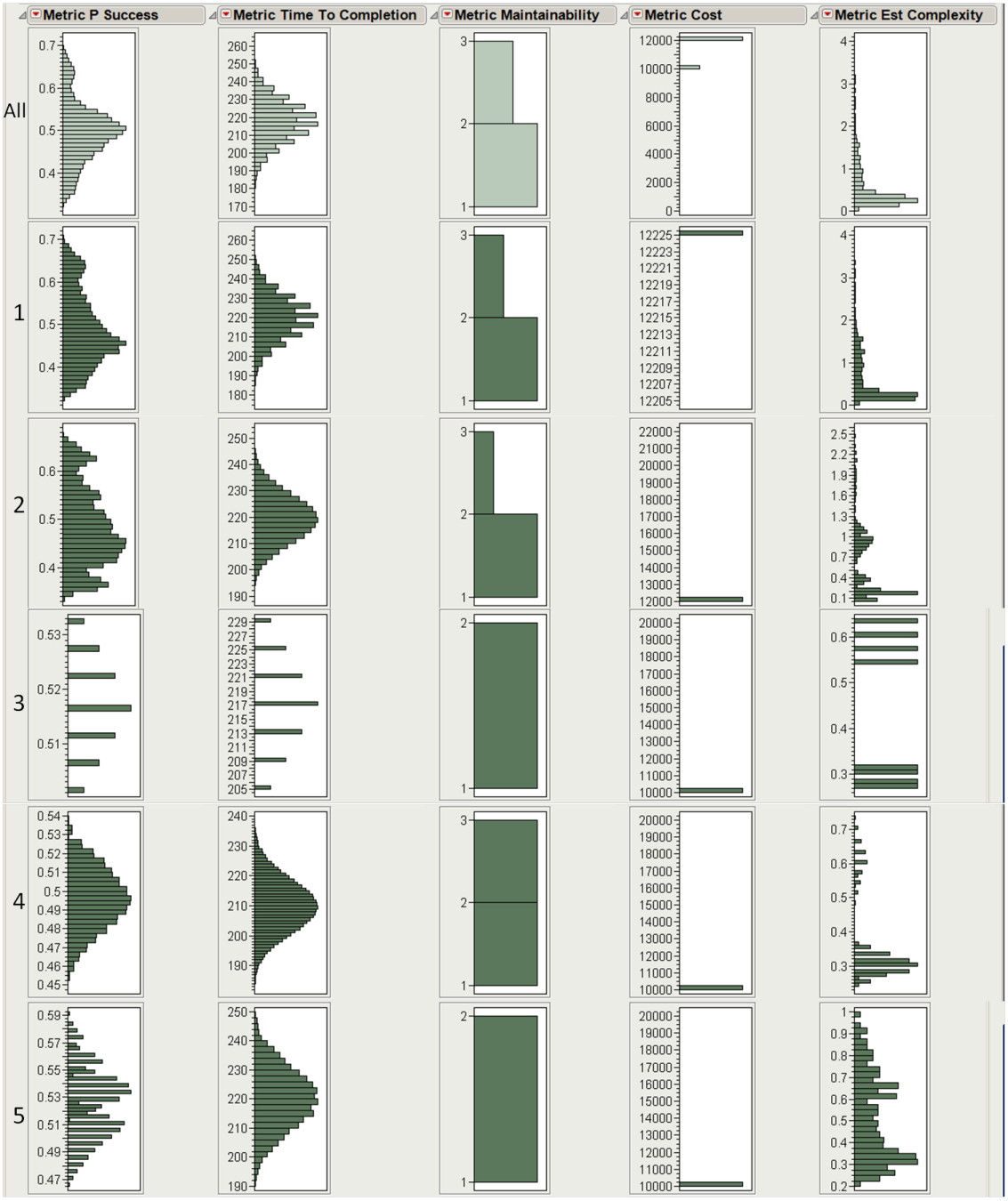


Figure 88: Distributions for the Portfolios 1 - 4

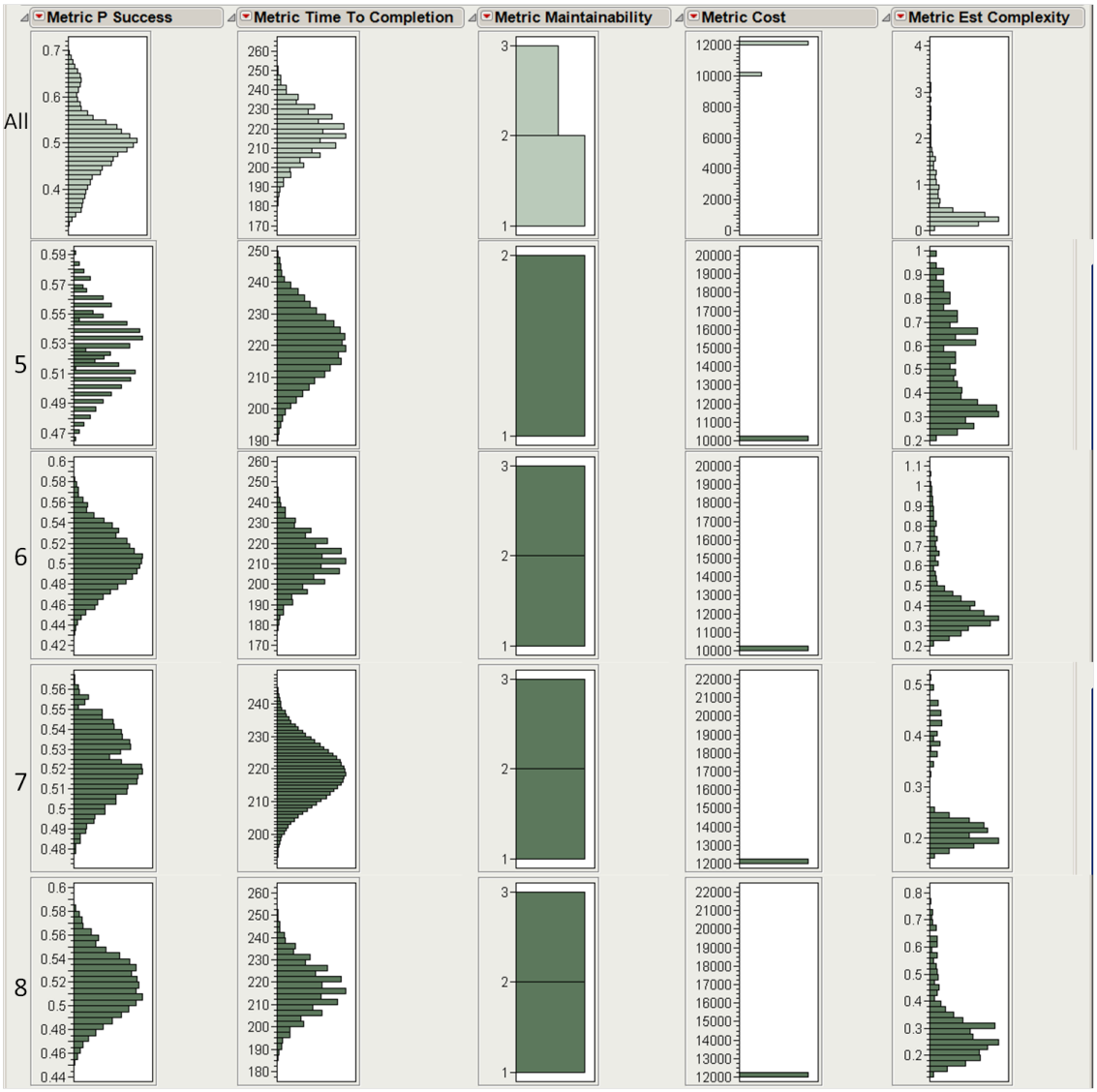


Figure 89: Distributions for the Portfolios 5 - 8

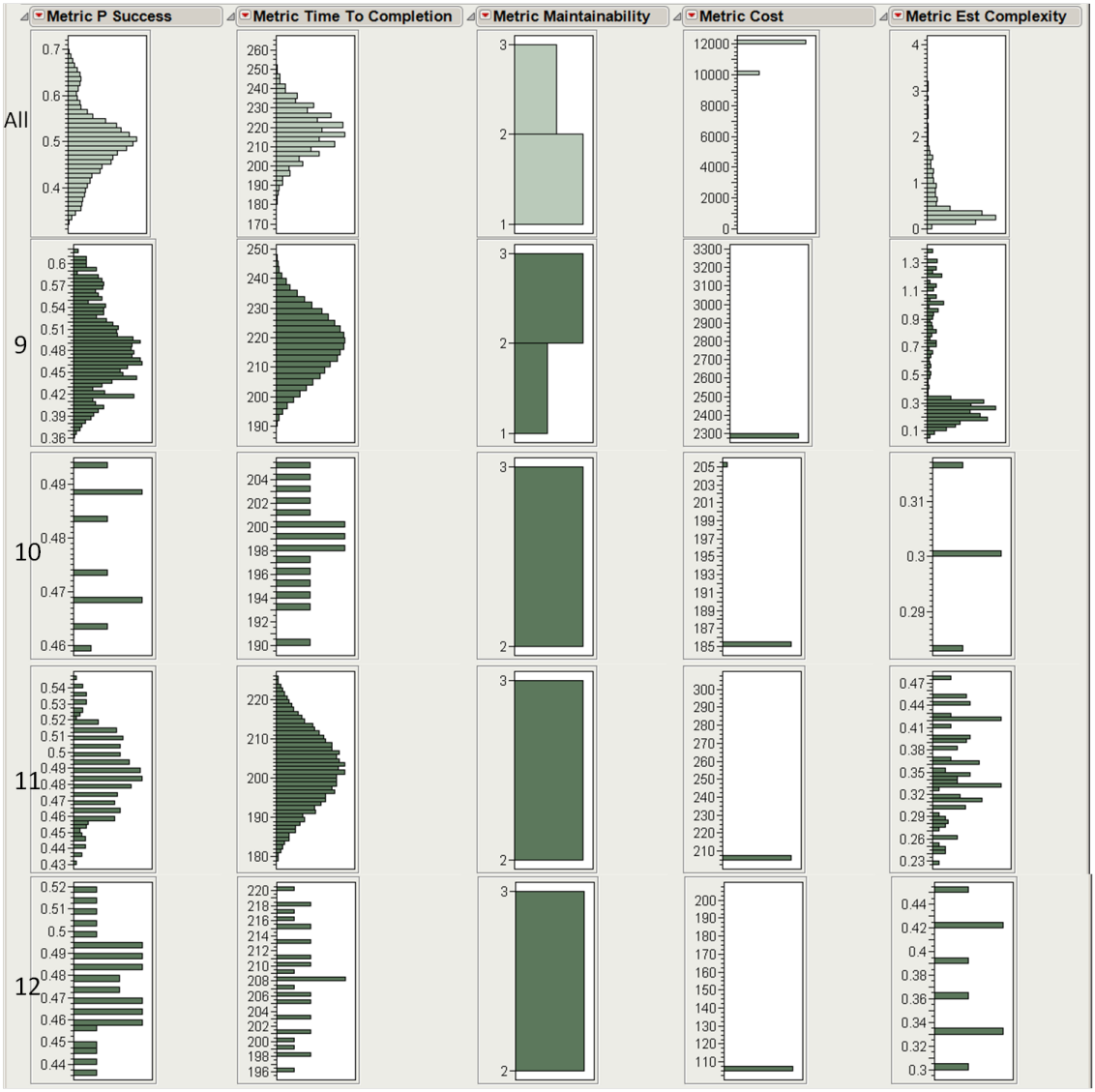


Figure 90: Distributions for the Portfolios 9 - 12

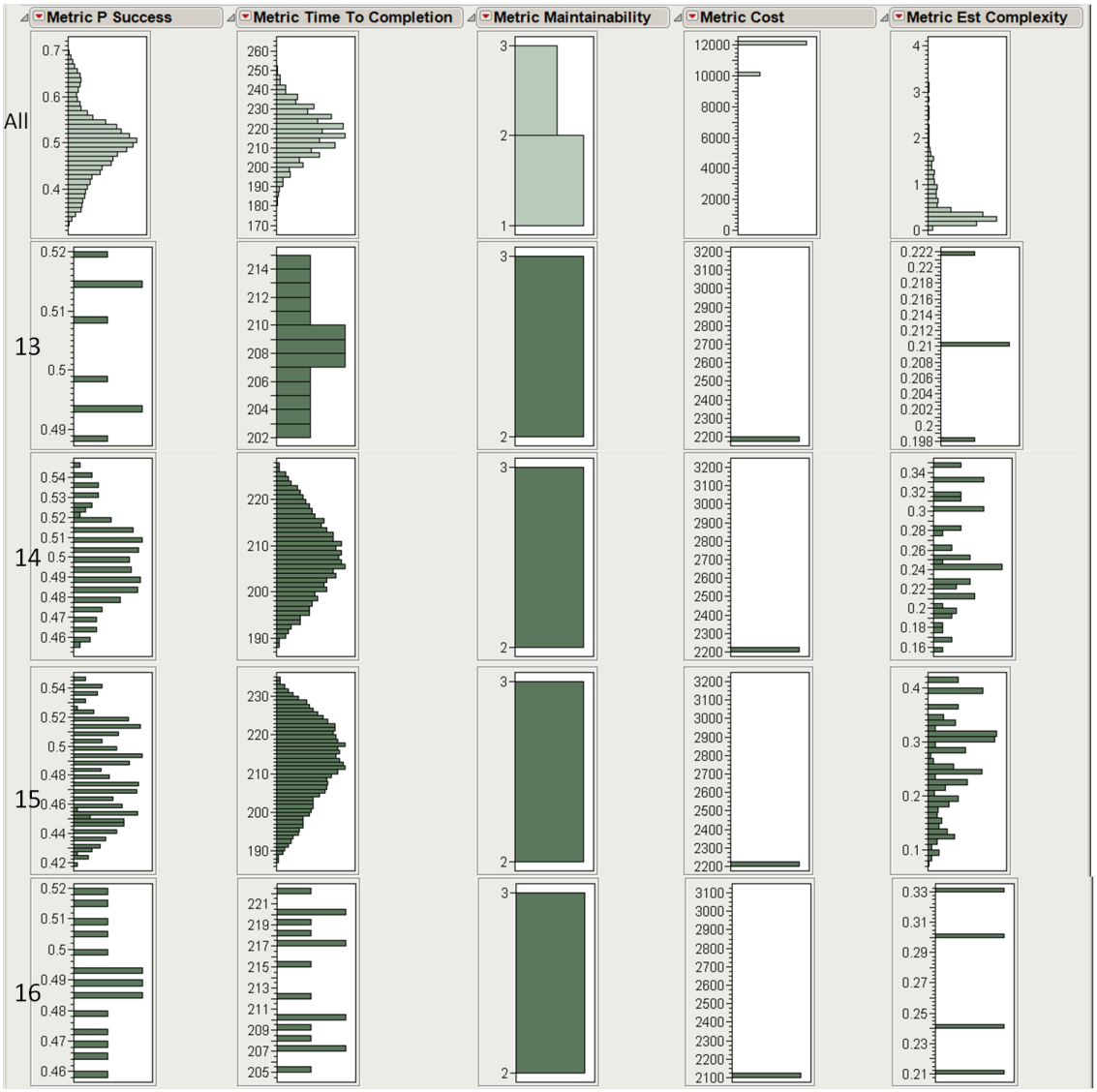


Figure 91: Distributions for the Portfolios 13 - 16

From the distributions, it is interesting to note the multi-modal behavior with respect to the probability of success for some of the portfolios, resulting in an overall multi-modal behavior for the alternative space. In order to discern the cause of this, the cases which are the in the smaller, better-performing mode are selected and compared to understand what is unique about those cases. Doing this reveals several things these cases are observed to have in common. First, an AH-64 is always selected to perform the task discriminate decoys. A DDG is always selected to engage to destroy. The wide area search is always performed by the EA-6B. This suggests that having these systems perform these tasks may give an increase in performance. There are two possible causes for this. First, that these tasks are so critical to the mission that any performance gains by one system over another have a significant effect. Alternately, it may mean that these systems are so far superior to other candidate systems for these tasks that using any other system drives down performance. With respect to the destructive engagement, the DGG is able to fire a large number of tomahawk missiles from outside of the active engagement zone, leading to an increased probability of success from both the perspective of increasing the likelihood of destroying targets and reducing the probability of being lost itself. However, for decoy discrimination, the AH-64 does not have a remarkably clear advantage over other assets. In this case, it is believed that the task itself is very critical to the mission, as failure in this task results in engaging the wrong targets, thus decreasing engagement success and increasing the probability of losing assets. The advantage gained by the low-flying AH-64 then becomes significant to the overall mission success.

In order to explore this further, filtering was applied to examine only those cases which meet each of the aforementioned conditions in turn. First, only those cases in which the AH-64 performs the task discriminate decoys were selected. The resulting distributions are shown in Figure 96, where the light green shows the distribution of all cases and the dark green shows those cases in which the AH-64 performs the

Table 12: Summary of Results for 16 SEAD Portfolios

Portfolio Number	Number of Alts	Mean P Success	Mean Time	Mean Maintainability	Mean Cost	Mean Complexity	Std Dev P Success	Std Dev Time	Std Dev Maintainability	Std Dev Cost	Std Dev Complexity
1	702408	0.49	219.77	1.32	12225.00	0.64	0.09	11.26	0.47	0.02	0.64
2	77279	0.48	219.04	1.23	12205.00	0.61	0.08	8.78	0.42	0.00	0.51
3	16	0.52	217.00	1.00	10170.00	0.44	0.01	6.53	0.00	0.00	0.15
4	5152	0.50	209.51	1.50	10185.00	0.35	0.02	8.56	0.50	0.00	0.12
5	976	0.53	219.62	1.00	10190.00	0.51	0.02	10.20	0.00	0.00	0.20
6	291872	0.50	211.67	1.50	10205.00	0.42	0.03	11.71	0.50	0.00	0.16
7	5152	0.52	218.51	1.50	12185.00	0.24	0.02	8.56	0.50	0.00	0.08
8	143360	0.52	217.08	1.50	12205.00	0.30	0.02	11.21	0.50	0.00	0.12
9	5275	0.49	217.44	1.67	2285.00	0.45	0.06	10.32	0.47	0.00	0.36
10	17	0.48	198.47	2.00	186.18	0.30	0.01	4.03	0.00	4.85	0.01
11	975	0.49	201.86	2.00	205.00	0.36	0.02	8.89	0.00	0.00	0.06
12	32	0.48	208.00	2.00	105.00	0.38	0.02	6.55	0.00	0.00	0.05
13	16	0.50	208.00	2.00	2185.00	0.21	0.01	3.50	0.00	0.00	0.01
14	480	0.50	207.28	2.00	2205.00	0.26	0.02	8.23	0.00	0.00	0.05
15	1066	0.48	212.37	2.00	2205.25	0.27	0.03	10.00	0.00	0.00	0.09
16	16	0.49	213.50	2.00	2105.00	0.27	0.02	5.59	0.00	0.00	0.05

discriminate decoys task. As can be seen from the figure, having this system perform this task causes a higher probability of success than the general case, and thus is a driver for mission success. Next, all of the cases where the DDG was selected for the engage to destroy task were selected, and the resulting distributions are shown in Figure 97, with the dark green representing the cases where the DDG was selected for engage to destroy. As can be seen from this figure, this selection did not have as large of an impact on the overall distribution, but is necessary for being in the higher mode of the probability of success distribution. Finally, all of the cases where wide area search is performed by the EA-6B were selected. The results, shown in Figure 98, show a similar trend to the DDG results, where there is not a large overall impact on the distribution shape in general, but it is necessary to be the better performing mode of the probability of success distribution. Since there is a desire to be in the high mode for probability of success, filtering can be applied to reduce the alternative space either to only cases in the top mode, or to only cases which meet these three criteria of system-to-task mappings. In this case, the latter will be done.

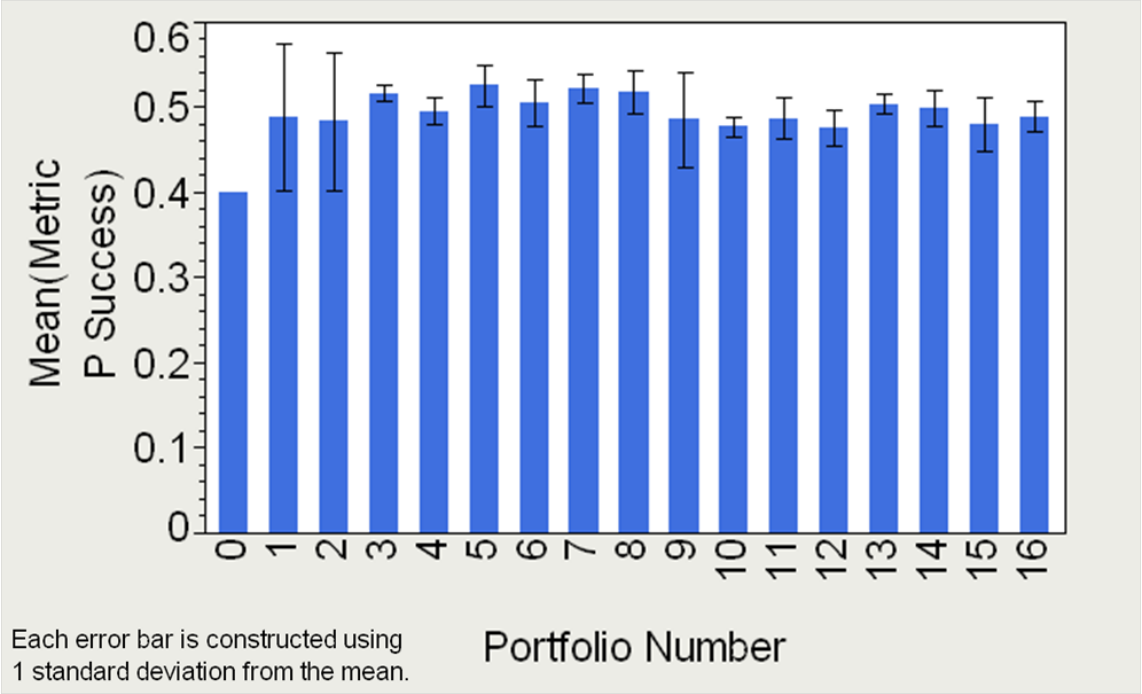


Figure 92: Mean Probability of Success for 16 Portfolios with Error Bars for 1 Standard Deviation

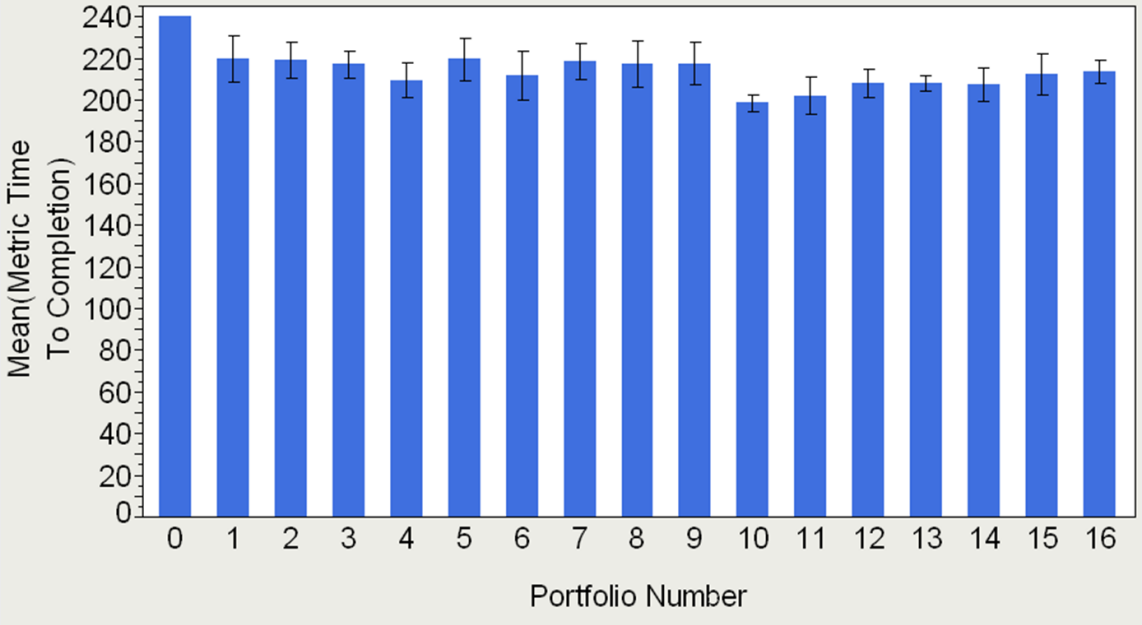


Figure 93: Mean Time for 16 Portfolios with Error Bars for 1 Standard Deviation

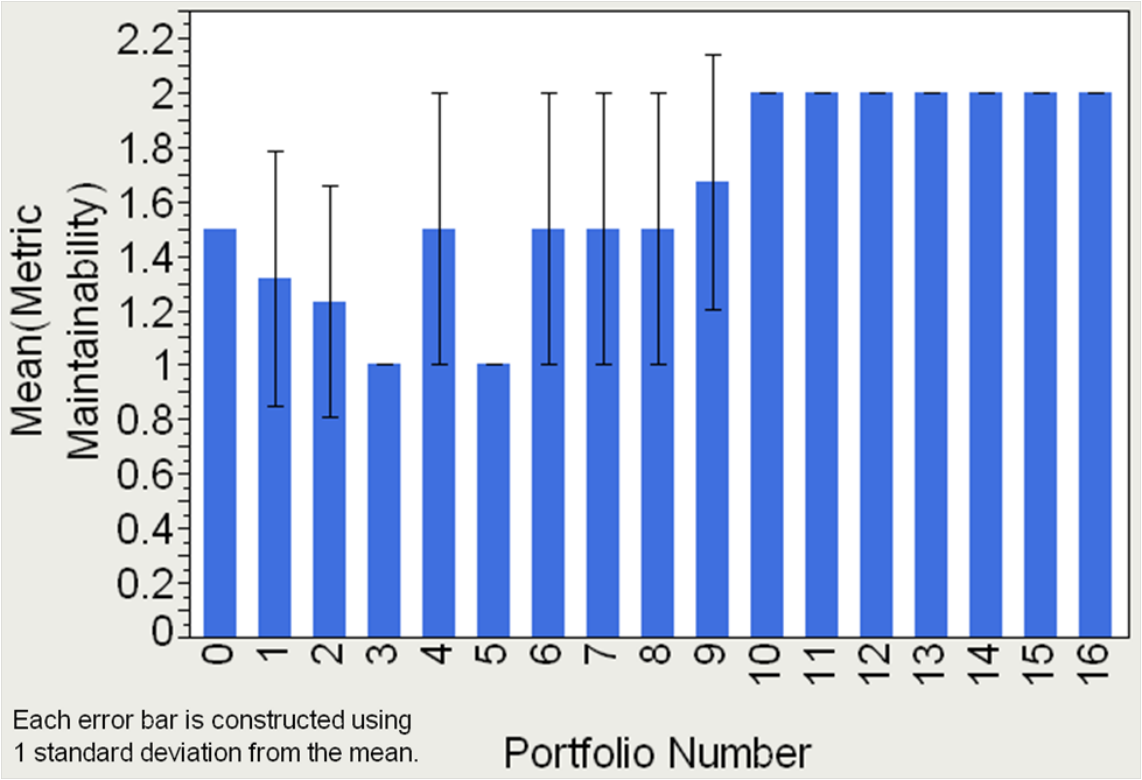


Figure 94: Mean Maintainability for 16 Portfolios with Error Bars for 1 Standard Deviation

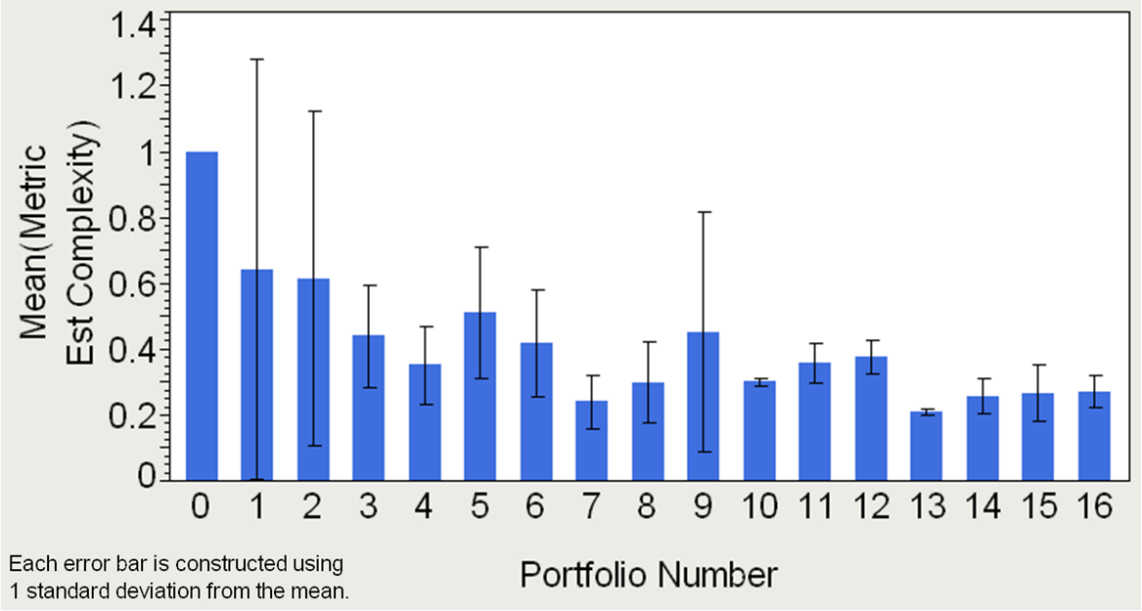


Figure 95: Mean Complexity for 16 Portfolios with Error Bars for 1 Standard Deviation

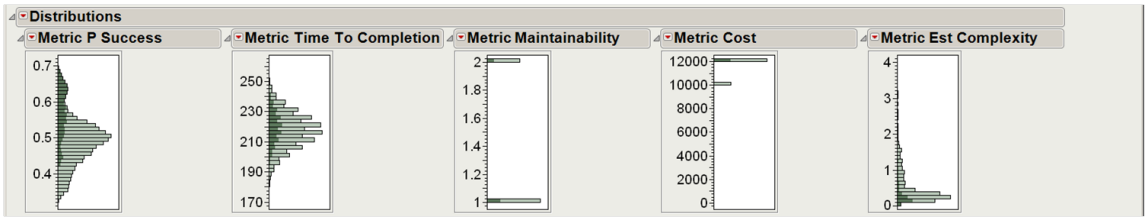


Figure 96: Results Distributions for Cases where AH-64 Performs Discriminate Decoys

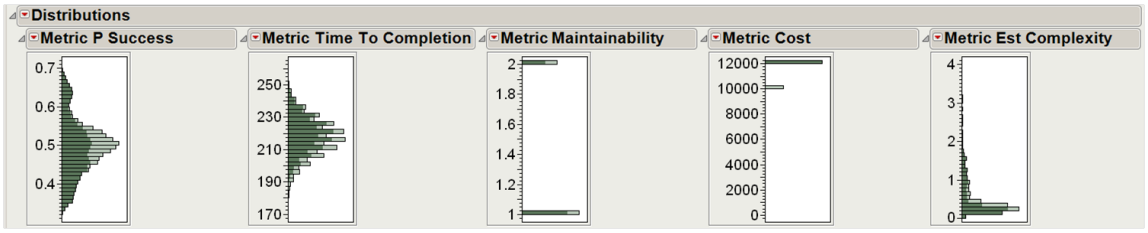


Figure 97: Results Distributions for Cases where DDG Performs Engage to Destroy

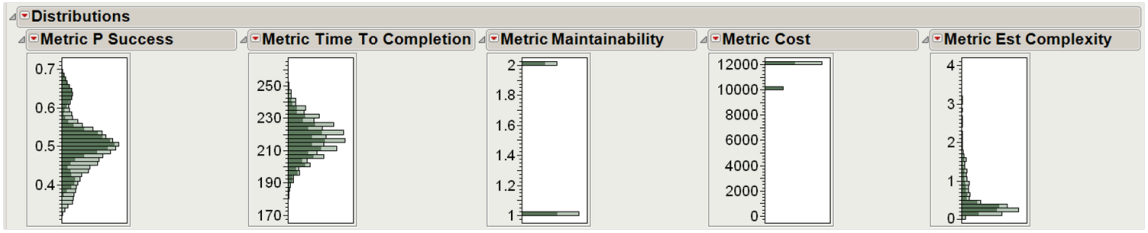


Figure 98: Results Distributions for Cases where EA-6B Performs Wide Area Search

The downselection described above leads to the remaining cases shown in Figure 99. The remaining points form two clear groups, of which the group with the higher probability of success has been highlighted in the Figure. It is of interest to determine the cause of this further grouping, so the selected points are again explored. For these selected cases, several observations are made. First, none of these cases use the AH-64 to perform target identification. However, they all include the AH-64, the central C2, the CVN, the DDG, the E-2, the EA-6B, and do not include the Mortar or the X-47b. As the application of these systems to tasks is still widely varied, these observations do not significantly help the downselection process.

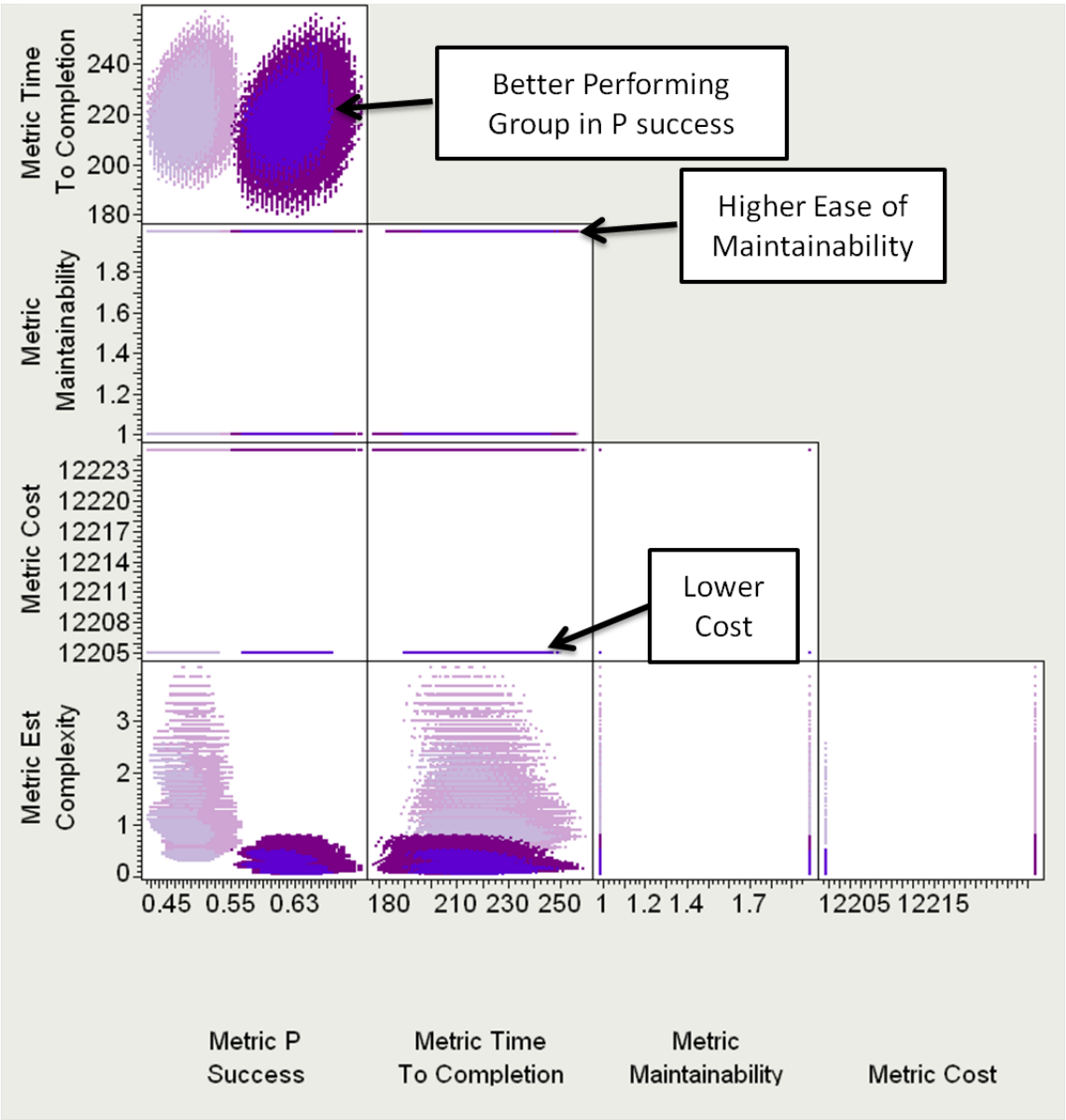


Figure 99: Resulting Scatterplot Matrix after System-Task Filtering

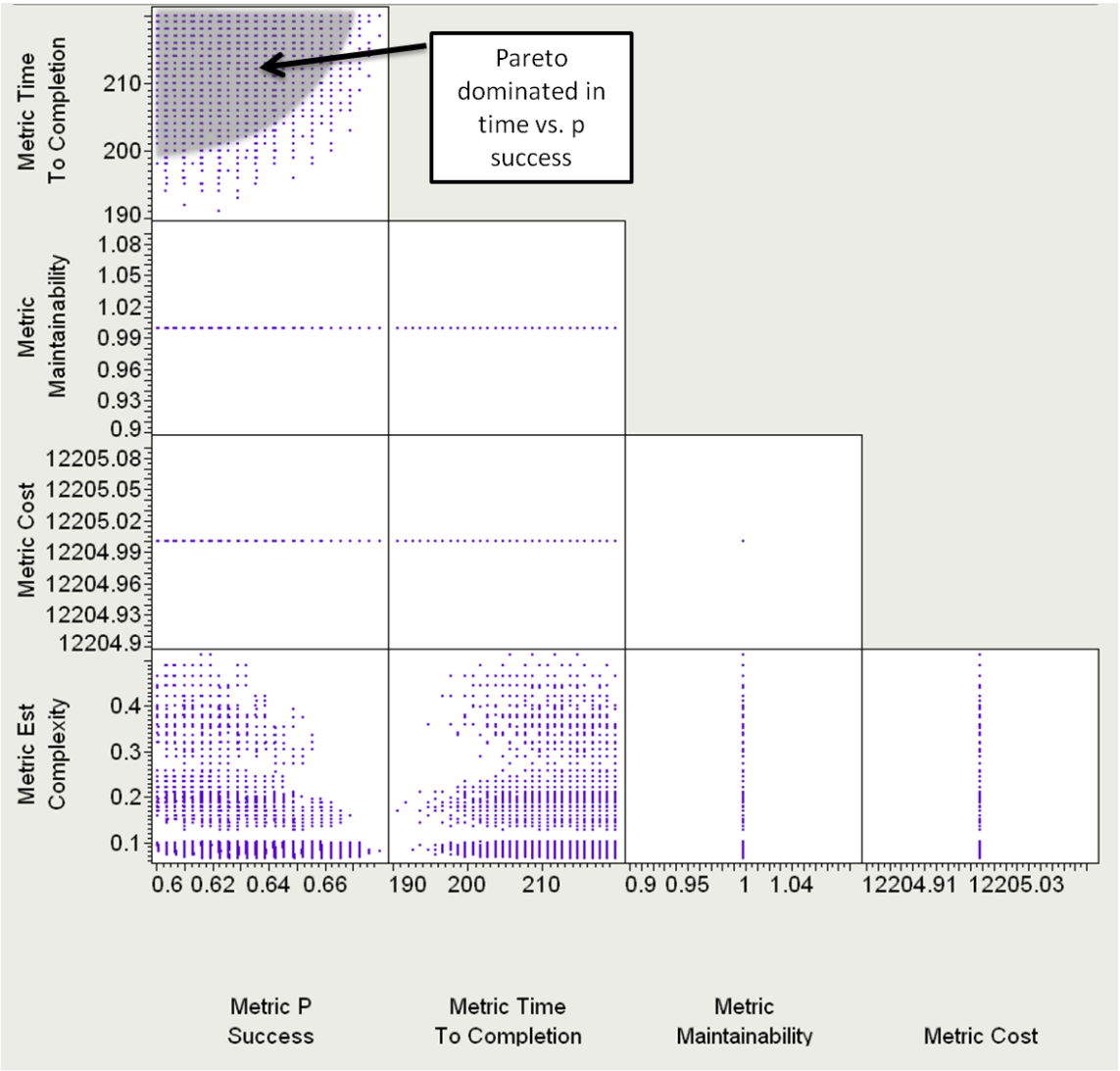


Figure 100: Resulting Scatterplot Matrix after Cost and Maintainability Filtering

Since applying only the three system-to-task mappings identified above did not significantly aid the downselection, the other approach was instead used, and all of the points not in the upper mode of the probability of success distribution were eliminated. In addition, a constraint of 220 minutes was put on the time to complete the mission. Applying these filters resulted in two clear groups of cases for both cost and maintainability. Filtering those cases with higher cost and less ease of maintainability resulted in the remainder of cases shown in Figure 100. These remaining cases show a clear Pareto frontier between time to completion and probability of success. As such, it makes sense to eliminate those cases that are Pareto dominated and carry forward only those that are on the Pareto frontier. Since there is uncertainty in these estimates, a band of cases along the edge of the Pareto Frontier are kept. The remaining cases are shown in Figure 101. Examining these remaining points further reveals that they are all instances of Portfolio 2. All of them use the CVN to assess engagement capability. All use the EA-6B to perform disruptive engagement in addition to the destructive engagement performed by the DDG. All use the AH-64 for discriminating decoys. All use the EA-6B to identify targets. All use the Central C2 to task sensors, and the E2 to track targets. The EA-6B is always used for wide area search. The X-47B is not included in these portfolios, indicating that the gaps in SEAD can be closed without pursuing a materiel solution, and that DOT_LPF changes should instead be recommended. Among these DOT_LPF changes, it is clear that using disruptive and destructive techniques together (which was not done in the baseline), is an important change. Furthermore, this analysis suggests that a ship which is sitting off-shore and launching tomahawk missiles into the theater (the DDG), and which is not in the combat zone is a better choice for destructive engagement. This represents a major operational shift over the baseline, where F-18 aircraft were used to fly into the theater and engage targets. The aircraft used in these remaining cases use the

AH-64 helicopter over a fighter jet or UAV, and use them primarily to help discriminate targets from decoys. Because the AH-64 would be able to fly fairly low, it makes sense that they could perform that task well. In 83 of the 109 Pareto frontier cases, the AH-64 is also chosen for battle damage assessment, for similar reasons. In 78 of the 109 cases, the E2 is chosen for sensor data fusion. In 108 of the 109 cases, the E2 is chosen to manage target movement data. Since so many of the cases use these three system-task pairs, they are added as filters to further reduce the alternative space. This results in the 58 cases shown in Figure 102.

The assumption will then be made that since all of the remaining alternatives meet the threshold to close the gap for time, priority will be placed on maximizing the probability of success, and thus the cases at the top end for probability of success are selected to be carried forward. The 11 alternatives chosen to move forward are shown in Figure 103. The task-system mappings for each of the alternatives are shown in Tables 13 and 14. A summary of the performance of these alternatives is shown in Table 15. Again, it should be noted that the decision process described above is one possible path through the downselection process. A decision-maker with different priorities may have made a different set of selections or given preference to different performance criteria.

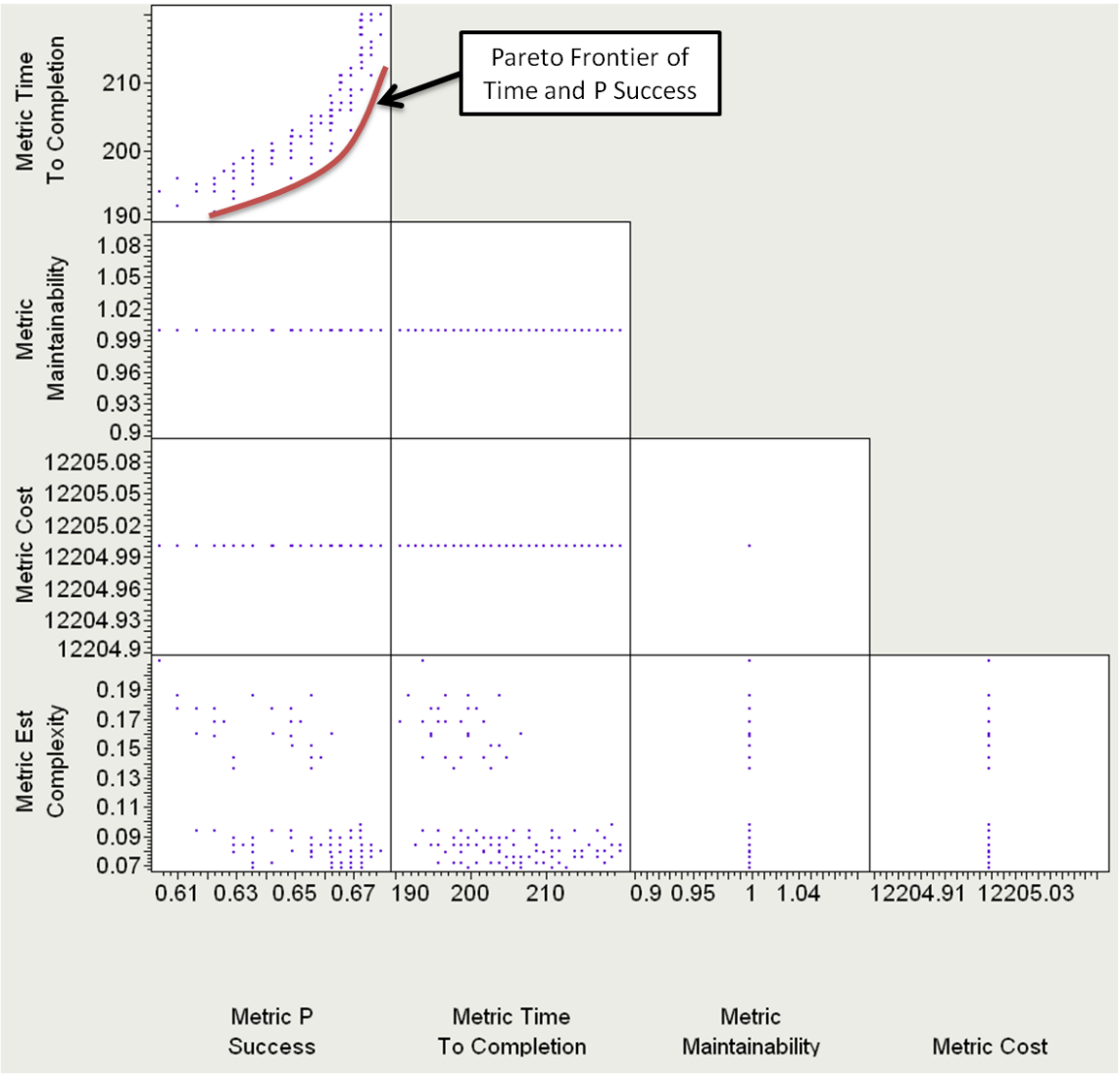


Figure 101: Pareto Frontier for Time and Probability of Success

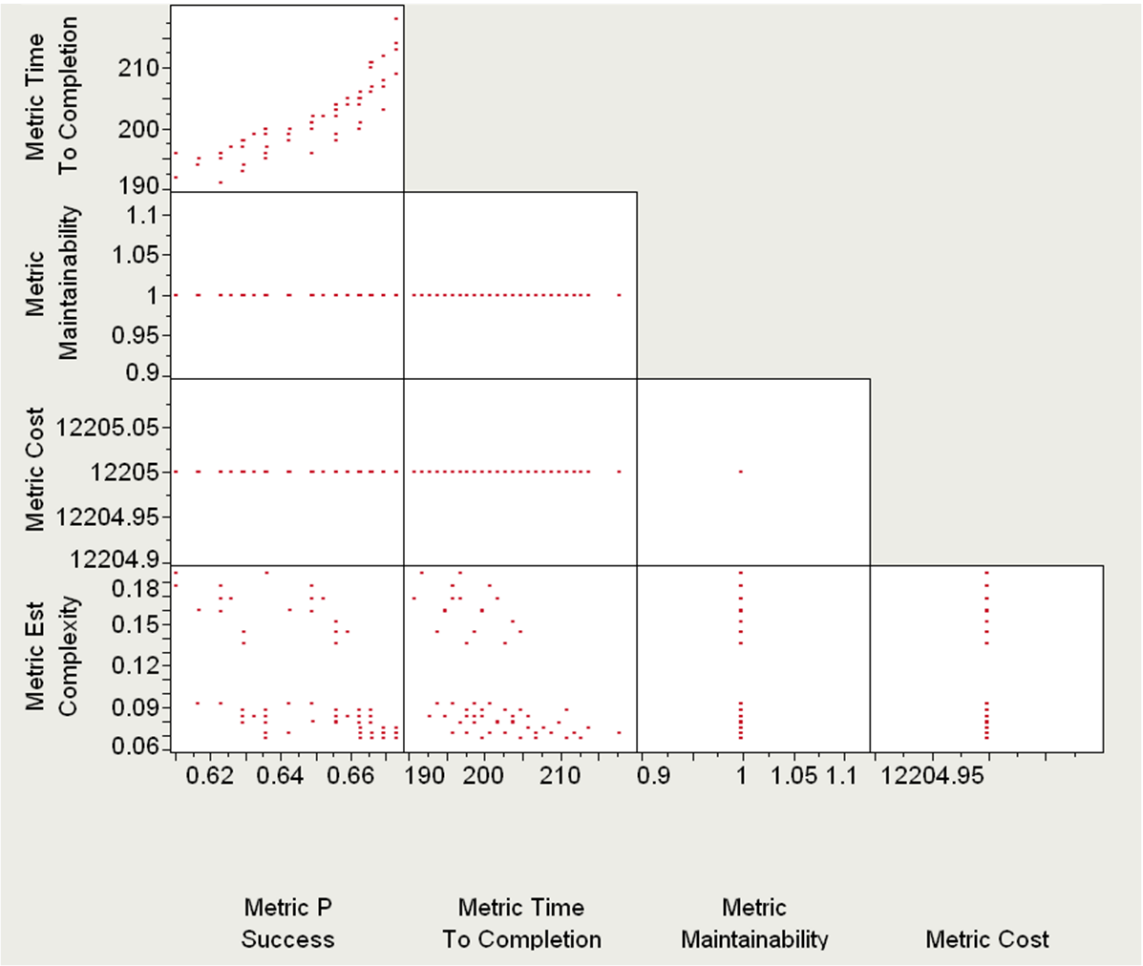


Figure 102: Remaining 58 Alternatives after Additional Task Filtering

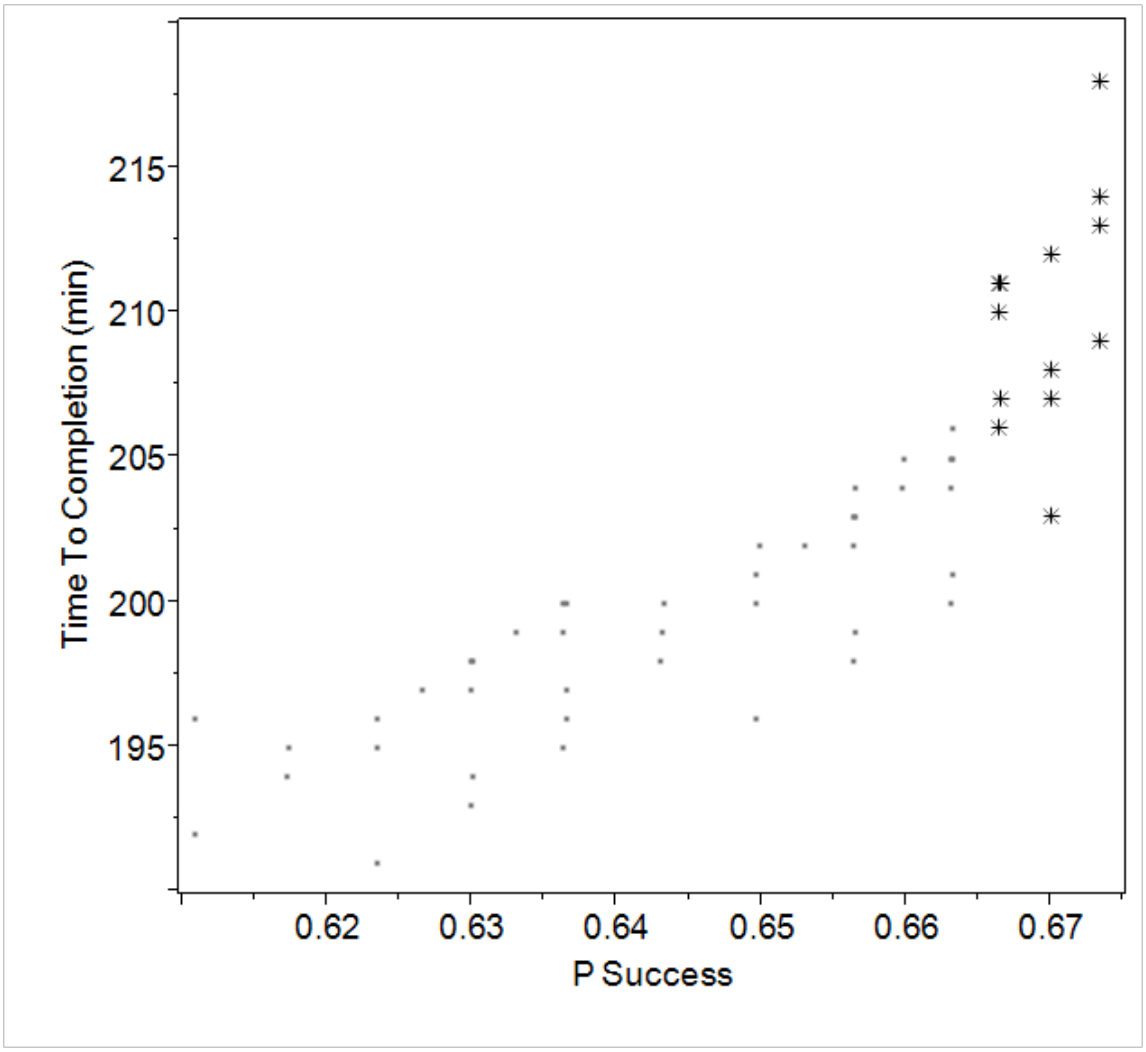


Figure 103: Final 11 Architecture Alternatives after Round 2 Downselection

5.6.3 Results from ARCNET and Engagement Model

These 11 finalists were then taken and run through ARCNET, which was linked with a simplified engagement model for SEAD developed by Domercant and documented in [59]. The scenario assumed in this model is shown in Figure 104. For each of the 11 finalists and the baseline, variations on the force structure, the enabled interfaces between systems (also called collaboration structure), and the interoperability level were made using a design of experiments. For each of the 11 alternatives, a 2-level full factorial DoE was used for force structure. For each alternative with each force structure, a fractional factorial was used for the collaboration structure and a 2-level fractional factorial was used for the IOL. For the baseline case, a three-level full factorial was used for the force structure. The baseline was done with three levels because it included an F-18, which had a wider range of force structure than the other assets. Since the range was larger, it was decided that using a middle level would be necessary to better understand the impact of force structure. The red force structure (which included early-warning radars, surface-to-air missiles (SAMs), and anti-aircraft artillery (AAA), was varied using a three-level DoE. Since the engagement model is stochastic in nature, each case was repeated 100 times, and the average and standard deviation across these repetitions was recorded.

Table 13: Summary of System-Task Mapping for 11 Finalists, part a

ID Number	Assess Engagement Capability	Assign Weapon And Platform	Battle Damage Assessment	Determine Sensor Availability	Discriminate Decoys	Engage To Destroy	Engage To Disrupt	Fuse Sensor Data	Wide Area Search
1	CVN	CVN	AH-64	Central C2	AH-64	DDG	EA-6B	E-2	EA-6B
2	CVN	CVN	AH-64	CVN	AH-64	DDG	EA-6B	E-2	EA-6B
3	CVN	CVN	AH-64	Central C2	AH-64	DDG	EA-6B	E-2	EA-6B
4	CVN	CVN	AH-64	CVN	AH-64	DDG	EA-6B	E-2	EA-6B
5	CVN	CVN	AH-64	Central C2	AH-64	DDG	EA-6B	E-2	EA-6B
6	CVN	CVN	AH-64	CVN	AH-64	DDG	EA-6B	E-2	EA-6B
7	CVN	Central C2	AH-64	Central C2	AH-64	DDG	EA-6B	E-2	EA-6B
8	CVN	Central C2	AH-64	CVN	AH-64	DDG	EA-6B	E-2	EA-6B
9	CVN	Central C2	AH-64	Central C2	AH-64	DDG	EA-6B	E-2	EA-6B
10	CVN	Central C2	AH-64	CVN	AH-64	DDG	EA-6B	E-2	EA-6B
11	CVN	CVN	AH-64	Central C2	AH-64	DDG	EA-6B	E-2	EA-6B

Table 14: Summary of System-Task Mapping for 11 Finalists, part b

ID Number	Manage Target Movement Data	Pass Warning And Location Data	Reconcile Target Priorities	Remove From Target List	Task Sensor	Track Until Stopped	Update Target List	Identify
1	E-2	Central C2	CVN	CVN	Central C2	E-2	CVN	EA-6B
2	E-2	Central C2	Central C2	CVN	Central C2	E-2	CVN	EA-6B
3	E-2	E-2	CVN	CVN	Central C2	E-2	CVN	EA-6B
4	E-2	E-2	Central C2	CVN	Central C2	E-2	CVN	EA-6B
5	E-2	Central C2	CVN	CVN	Central C2	E-2	Central C2	EA-6B
6	E-2	Central C2	Central C2	CVN	Central C2	E-2	Central C2	EA-6B
7	E-2	Central C2	CVN	CVN	Central C2	E-2	CVN	EA-6B
8	E-2	Central C2	Central C2	CVN	Central C2	E-2	CVN	EA-6B
9	E-2	E-2	CVN	CVN	Central C2	E-2	CVN	EA-6B
10	E-2	E-2	Central C2	CVN	Central C2	E-2	CVN	EA-6B
11	E-2	E-2	CVN	Central C2	Central C2	E-2	CVN	EA-6B

Table 15: Summary of Performance for 11 Finalists

ID Number	Metric P Success	Metric Complexity	Metric Time To Completion	Metric Maintainability	Metric Cost	Metric Est Complexity
1	0.67	216	209	1	12205	0.07
2	0.67	204	213	1	12205	0.07
3	0.67	228	214	1	12205	0.08
4	0.67	216	218	1	12205	0.07
5	0.67	216	207	1	12205	0.07
6	0.67	204	211	1	12205	0.07
7	0.67	216	203	1	12205	0.07
8	0.67	204	207	1	12205	0.07
9	0.67	228	208	1	12205	0.08
10	0.67	216	212	1	12205	0.07
11	0.67	266	211	1	12205	0.09

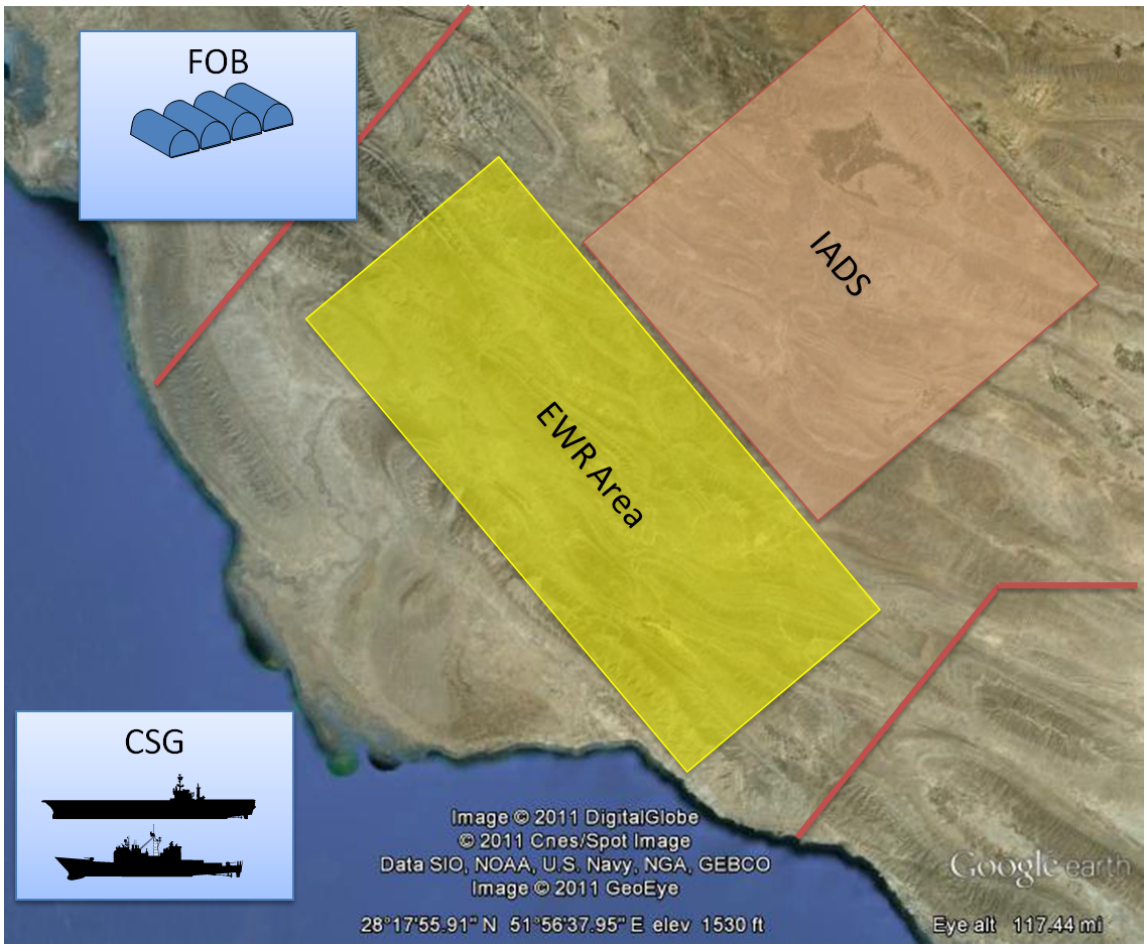


Figure 104: Scenario used in Simplified Engagement Model Developed by Domercant for ARCNET. Adapted from [59]

For the baseline, the force structure was found to play a more significant role in the performance than the collaboration structure or the IOL, although the IOL had a noticeable effect on performance for small force structures. These results are depicted in Figure 105. The figure shows two plots. The first plot shows the average percent of red units suppressed against the average number of blue units lost. On this plot, the most desirable alternatives will be located in the upper left-hand corner, where the smallest number of units are lost and the greatest number of units are suppressed. The second plot shows the average percent of red units suppressed against the RPC for each alternative. The RPC is influenced primarily by the IOL. Each force structure is shown using a different color and shape combination. Since the only asset in the baseline of which the quantity can be varied is the F-18, and a three-level DoE was used, there are three force structure alternatives. The circles represent two F-18s, the cross represents four F-18s, and the diamond represents 6 F-18s. The groupings of points by shape indicates that force structure is driving the performance variations. As is expected, increasing the number of F-18s increased the percentage of red units suppressed. However, it also results in a greater number of blue units destroyed. This is because the F-18s, even with 6 of them, are unable to find and destroy all of the early-warning radars in time, resulting in increased losses as more assets are added to the mix.

It is expected that increasing the force structure beyond what was considered for this study would result in improved performance with fewer losses, and in fact, this can be seen by re-running ARCNET with a larger allowable force structure (up to 9) for the F-18, as shown in Figure 106. This figure was created using a finer granulation on the collaboration structures in order to better show the effects of this parameter. In the figure, the circles represent a case with 3 F-18s, the crosses represent a case with 6, and the diamonds represent a case with 9. In the case with 9 F-18s, the quantity of F-18s overwhelms the enemy, resulting in improved performance with less

losses then the case with 6 F-18s. However, in this notional study, an assumption was made that only a small number of F-18s would be available to be deployed from the carrier for the SEAD mission because the rest would be deployed for the follow-up mission. This is the same assumption that was used in early assessments of the baseline performance and when making performance estimates in RAAM.

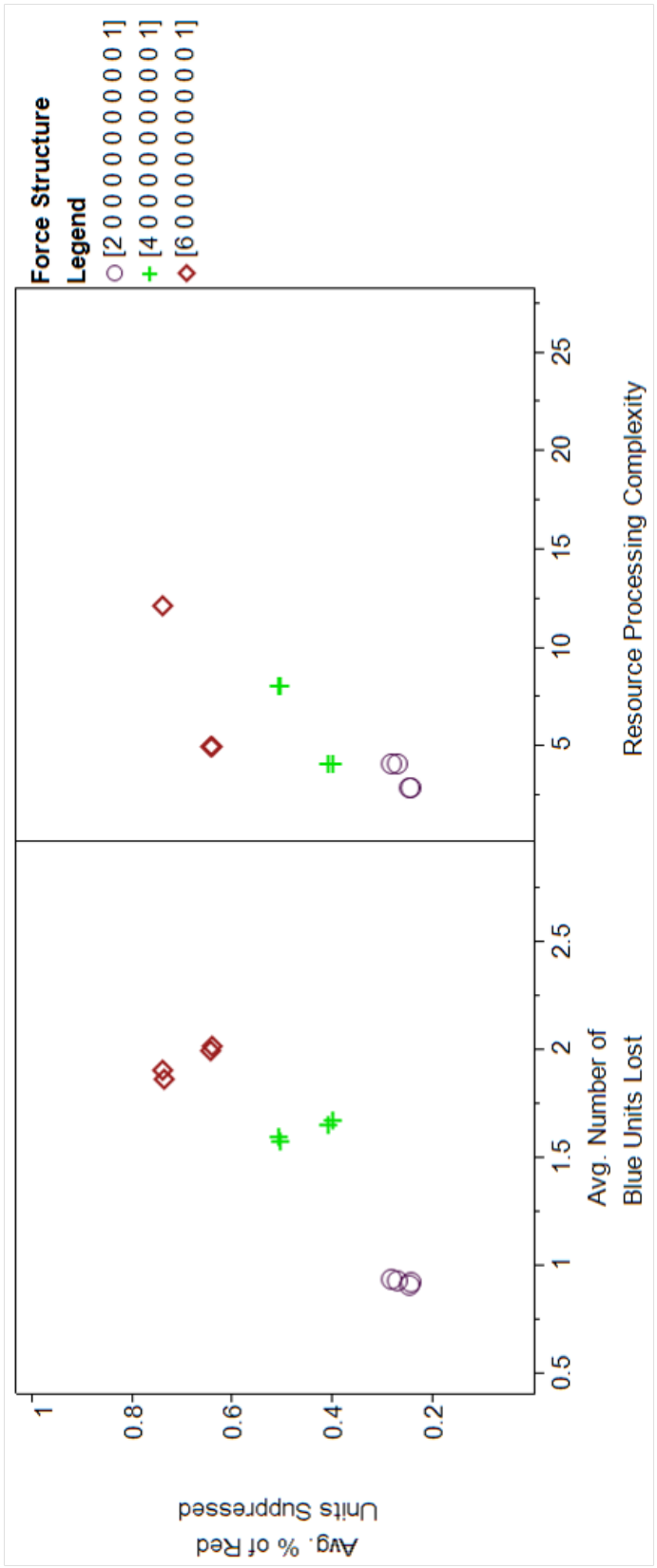


Figure 105: Baseline Performance in ARCNET

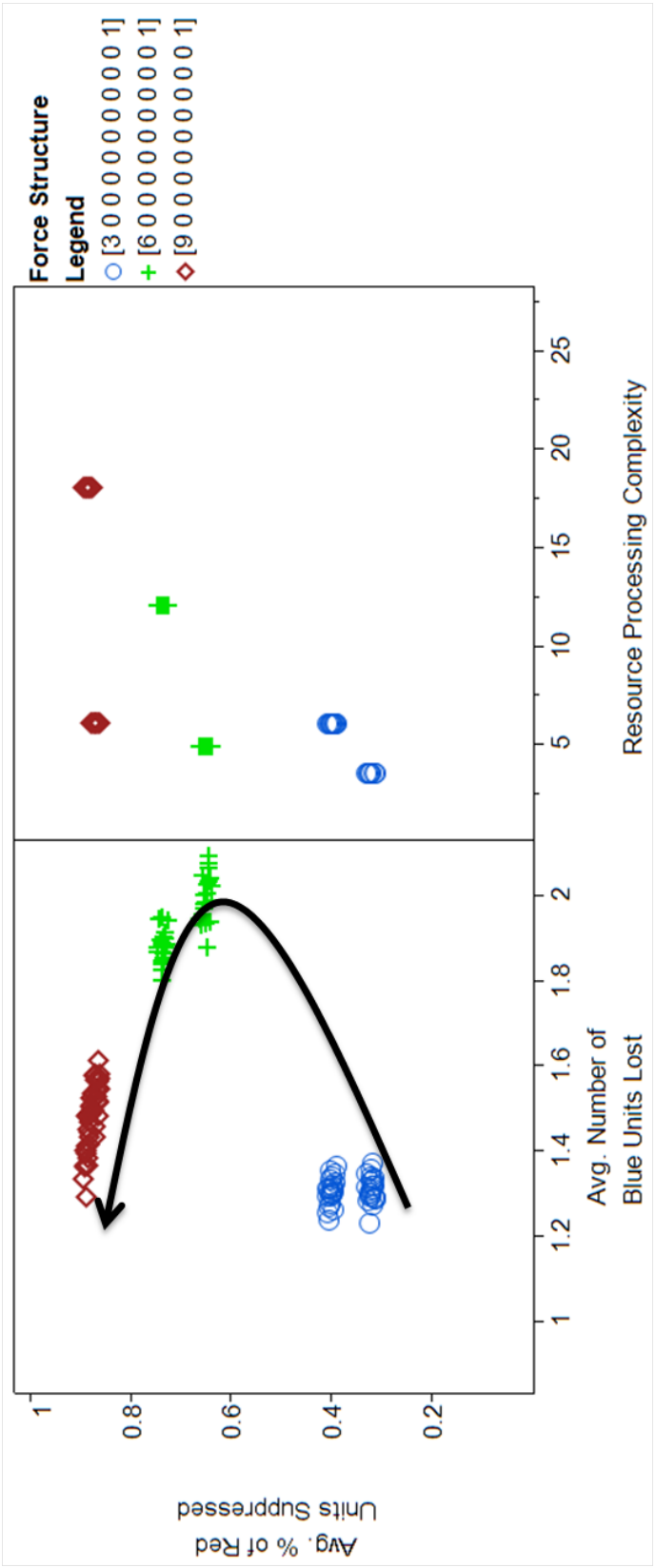


Figure 106: Baseline Performance in ARCNET with up to 9 F-18s

Although the force structure has the most noticeable impact on the overall performance, the IOL still had a noticeable impact within a given force structure. The RPC, one of the attributes of complexity proposed by Domercant in [59], gives an indication of the level of networking. Within a force structure alternative, the lowest RPC represents the minimal possible networking to complete the mission, while the highest RPC represents a maximally connected network. Since the magnitude of the RPC is dependent somewhat on the force structure (more assets can have more interfaces, resulting in a higher RPC), it is not the magnitude of the number that is of interest, but rather where the points fall relative to each other within a single force structure. This can be seen in the plot on the right-hand side, which plots the RPC against the number of successful engagements. If one imagines drawing a line connecting all of the points with the same shape (i.e. the same force structure), the slope of that line indicates the impact of increasing collaboration. If the line has a positive slope, then increasing collaboration increases performance, a negative slope indicates that increasing collaboration decreases performance, and no slope indicates that collaboration does not impact performance on the successful engagements metric. For the cases where there are less total assets, collaboration has a noticeably positive slope, indicating that increasing collaboration does improve performance. However, as the force structure includes more assets, the slope levels out such that there is not a noticeable impact on performance from increased collaboration.

On the plots of RPC against the average red suppressions, there are two distinct groups for each force structure, representing the high and low levels of IOL that were used in the two-level IOL DoE. There are multiple cases within each of these groups, representing the different collaboration structures for that IOL and force structure. The grouping of the points shows that collaboration structure has little effect on the overall percentage of units destroyed, although it has a small but noticeable effect on the number of blue losses. The results of increasing IOL are dependent on the force

structure. In Figure 105, there is a small increase in red suppressions when there are only 2 F-18s, with noticeable change in both red suppressions and blue losses for the cases with 4 and 6 F-18s. Figure 106 shows these effects with greater granularity. The case with 3 F-18s has a jump in percent of suppressed red units with the varying IOLs and a shift in losses with increased collaboration. The cases with 6 F-18s show a shift in both losses and suppressions for the IOL, and again, only in losses for the collaboration structure. However, once there are 9 F-18s, increasing the IOL no longer has any noticeable effect, although increasing collaboration can still help to reduce losses. In fact, the impact of collaboration increases as the number of forces increases. This is shown in Figure 107.

For the 11 alternatives that were carried through to this stage of the evaluation, the ARCNET results showed similar performance in the engagement. This was expected from the RAAM results, as the architectures selected to be carried forward for analysis all showed similar peformance across all metrics in RAAM. Since the 11 alternatives show similar performance levels and trends, the first alternative will be discussed here to demonstrate how a decision-maker might use the results of ARCNET, and the results for the remaining alternatives (which can be analyzed in a similar way) are shown in D. The results for alternative 1 are shown in Figure 108.

Similar to the baseline, the results show groupings in performance according to the force structure. The legend can be interpreted as a vector of all the possible systems included in the study with the integer value representing the number of that system included. The order of the systems is: F/A-18, AH-64, X-47B, EA-6B, Mortar, DDG, SOF, E-2, Intel Satellite, Central C2, and CVN. The relevant systems (those with a non-zero value) to this case are, in order: AH-64, EA-6B, DDG, E-2, Central C2, and CVN. These systems are used according to the alternative 1 system-task mappings shown in Table 13. While this alternative behaves similarly to the baseline with respect to the force structure groupings, the impact of IOL and collaboration

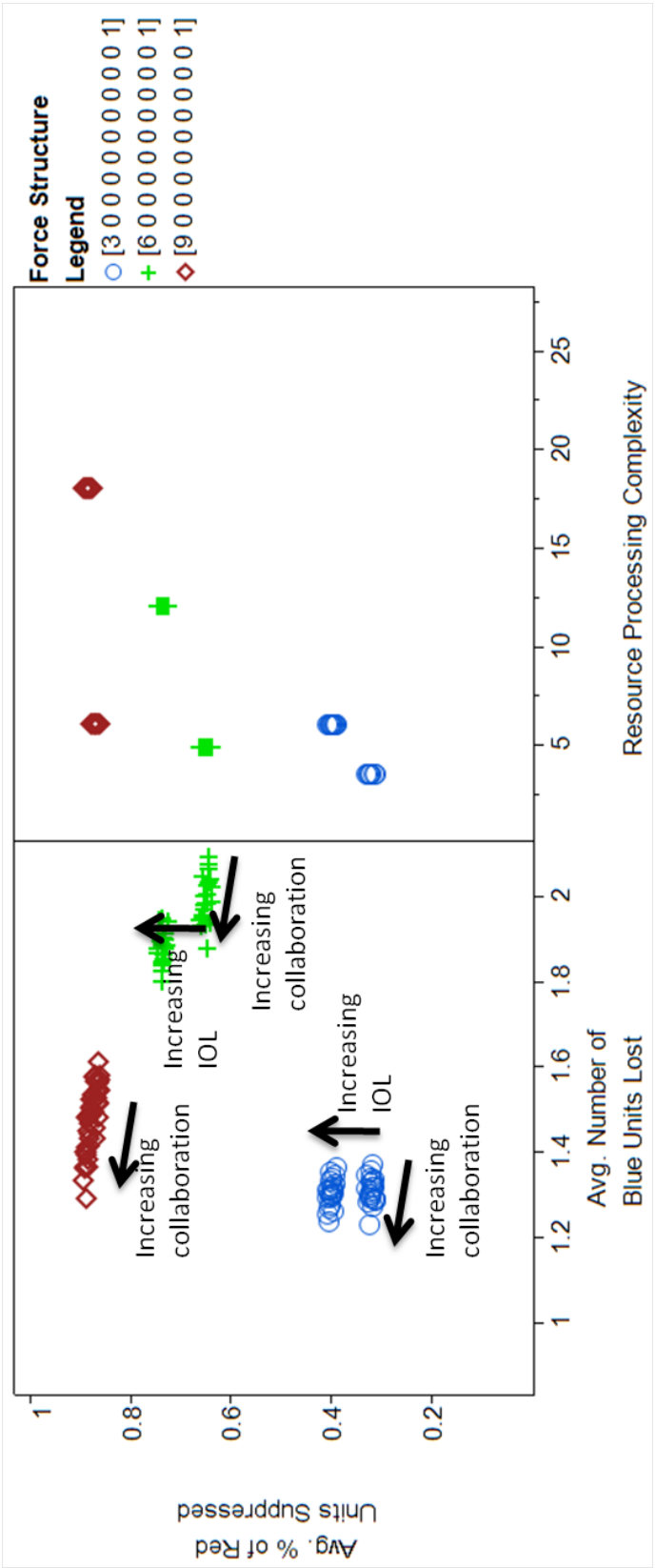


Figure 107: Impact of IOL and Collaboration Structure on Performance

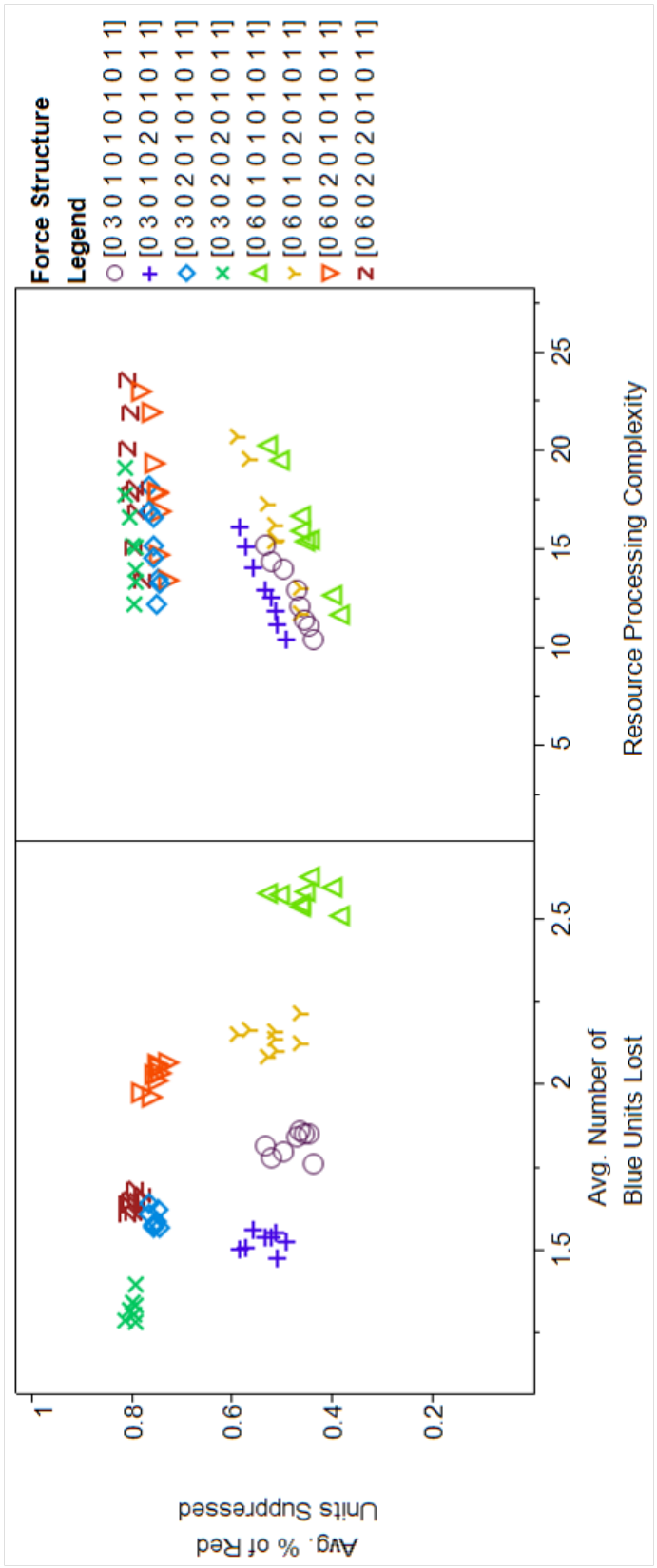


Figure 108: Alternative 1 ARCNET Results

structure is not as obvious. In fact, there is almost no noticeable impact from either one on blue losses, although for small force structures there is still a noticeable impact on the number of red suppressions. It is hypothesized that the reason for this is that the systems chosen to perform engagement (both disruptive and destructive, the DDG and EA-6B) are able to act against multiple targets at once, and are not at risk to be shot down during the engagement process. Thus, unlike the baseline where the F-18s rely on each other to obtain needed information to both locate targets and avoid detection, the set of systems used here does not need to rely on the networked effects as heavily.

There are two clear bands of force structures with respect to the average percentage of successful suppressions. The top band, which is highlighted in Figure 109, corresponds to those cases which include 2 (rather than 1) DDGs. This occurs because this doubles the rate at which tomahawk missiles can be fired, and thus greatly improves the performance against the targets. The best performing alternative uses a maximum number (2) of DDGs and EA-6Bs, which makes sense because it is able to most quickly find and engage targets. What is interesting to note however, is that it uses the lower number (3, as opposed to 6) AH-64s. Since these are the assets flying into the engagement zone and at risk of being shot down, it makes sense that the less there are, the less that will be hit. However, it is of great interest that having less of these assets does not result in a depredation in performance with respect to the number of targets suppressed. This implies that for the tasks being done by the AH-64 (battle damage assessment and decoy discrimination), 3 is enough and having more of this asset is unnecessary.

The aforementioned results may lead decision-makers to examine the effect of having only two or one of this asset. Having none is not an option since the mission would not longer be able to be fulfilled. In order to study this, ARCNET is again run, but with the DDG and EA-6B fixed at two assets apiece, and with the number

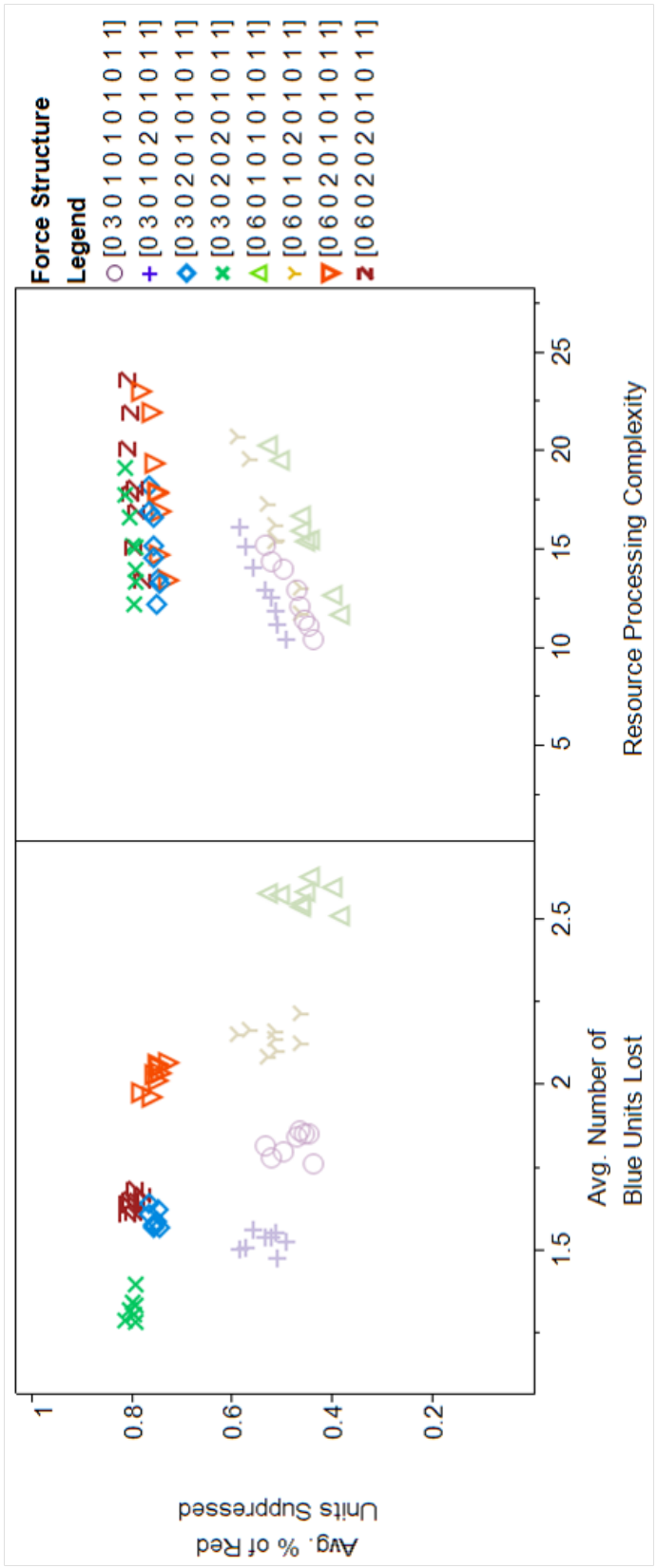


Figure 109: Alternative 1 ARCNET Results, with Upper Band Highlighted

of AH-64s varied from 1 to 3. Figure 110 shows the results of this study. There is no degredation in suppression performance as a result of reducing the number of AH-64s, but the average number of units lost does decrease. This shows that only one AH-64 is needed to perform battle damage assessment and decoy discrimination, and the use of more than one is unnecessary. Thus, it would be recommended to only use one AH-64 in performing SEAD with this architecture.

Looking at the second plot in Figure 108, it can be seen that for the top performing alternatives, increasing the complexity through increases to IOL and collaboration structure has very little effect on the ability to suppress targets. This is observed by the way each force structure alternative shows an almost flat line as RPC increases. In the cases where there are fewer DDGs, there is a greater effect from increasing the level of net-centricity. As the RPC increases, there is roughly a twenty percent improvement in the average percentage of red units suppressed.

The results for the remaining alternatives are very similar to the ones shown here, and can be analyzed in the same fashion. The main differences between the alternatives included in this final modeling step are in the C2 elements of the architecture (i.e. which C2 systems are assigned to which C2 tasks). Since the performance is similar for all, it can be concluded that as long as the C2 is sufficient to enable the mission to be completed, the way in which it is implemented has little impact on the mission outcome. This may not be true for other missions, such as close air support, where mission success is more heavily reliant on real-time C2.

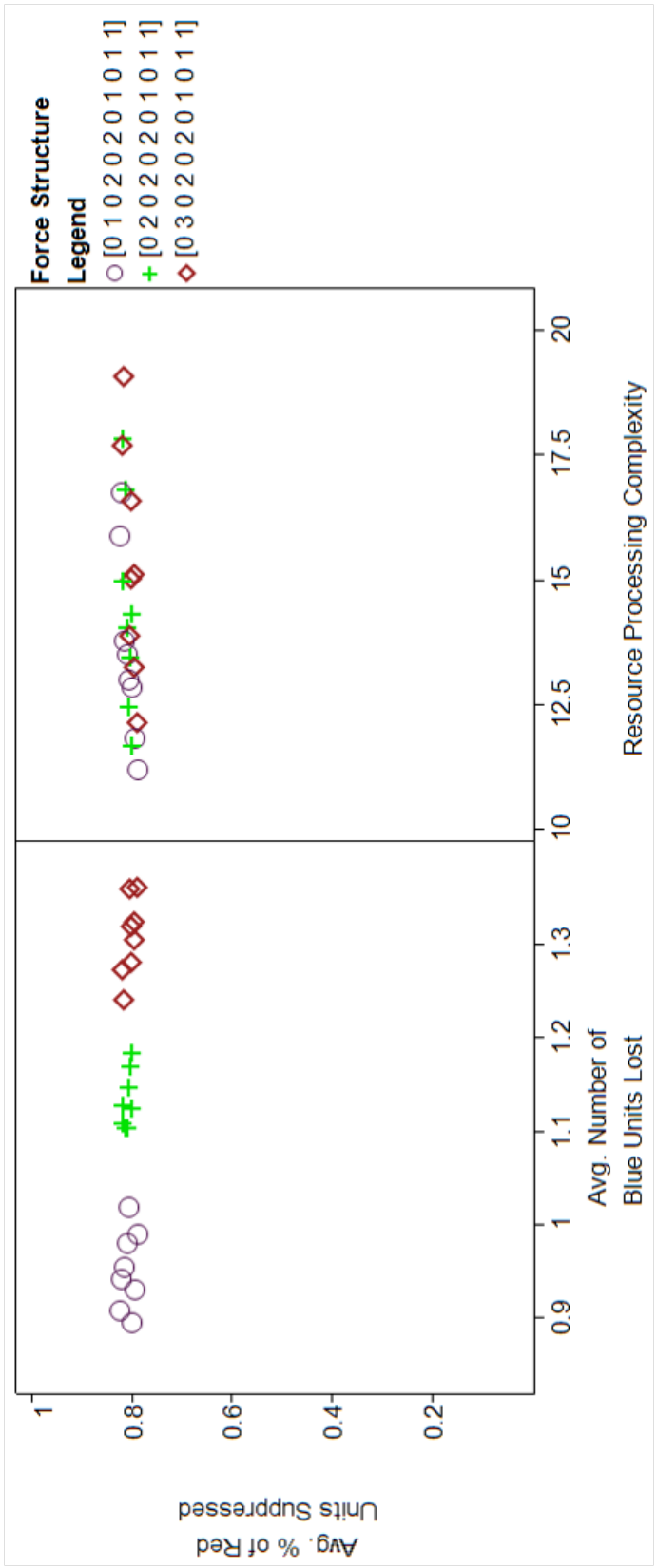


Figure 110: Alternative 1 ARCNET Results, with 1-3 AH-64s

5.6.4 Pre-Milestone A Decision-Making for SEAD

In the case of the SEAD example presented here, it is not necessary to employ the ARC-VM framework recommended in 4, because the remaining options do not include any new acquisitions. Thus, in this case, it would be recommended to decision-makers that for SEAD, it is unnecessary to invest in new systems when existing systems can be used in a new way to perform the mission. It would further be recommended that any further analysis to improve upon or verify these results focus more on the searching and engagement elements of the mission rather than on the C2 elements, as it has been shown that the C2 elements are not driving mission success. Decision-makers would also be advised to verify that the life-cycle of the selected systems is long enough that they will not have to be replaced in the near future, otherwise, it may again be necessary to explore new materiel solutions. However, if the decision is made to move down a materiel path for reasons other than performance, it would be suggested that alternate materiel solutions be suggested beyond the ones chosen for this study, and that the analysis be repeated with these other systems. This is because the materiel solutions selected for study here did not perform adequately for the SEAD mission. However, the results from this study can advise what the characteristics should be included for a new materiel solution for SEAD. It was observed that long-range, stand-off type engagement systems are more effective in this mission, and thus future materiel solutions should perhaps be of this type. This insight can be used to develop the requirements for a new research effort. Furthermore, it is suggested to decision-makers that tactics for SEAD should focus on effectiveness through numbers rather than through networked effects. In the case of this mission, it was shown that the number of forces, and in particular the number of engaging and sensing assets has the largest impact on mission success. It should be noted that the type of results given by this study are not typically included in previous CBA studies to date, which are typically heavily focused on materiel solutions, and the use of the ARCHITECT

method enabled a much more thorough CBA analysis than is traditionally done. However, it should not be forgotten that this method is helping to prune down the very large alternative space into a more manageable set of alternatives. More detailed analysis of these alternatives will be required to better study the implementation into the existing SoS and to better verify the expected performance outcomes predicted here.

5.7 Observations and Lessons Learned from SEAD Study

While the numerical results of the SEAD study are notional and could not be applied to an actual SEAD CBA, they nonetheless show interesting results and can be used to highlight the strengths of the ARCHITECT method, as well as some areas for future work. The lessons learned from this study also highlight some key considerations for the implementation of the methodology. Although many observation have been made throughout the presentation of the example, some of the key points will be discussed here.

First, and most importantly, the SEAD mission demonstrates the overall feasibility of the ARCHITECT method, and helps to substantiate many of the expectations of performance discussed in 4. The SEAD study here enabled pre-Milestone A trade-offs to be performed quantitatively across a large number of architectural alternatives in a traceable and repeatable manner. The alternatives considered included variations on operations, systems, organizational responsibilities (through the assignment of systems to tasks), network (or collaboration) structure, interoperability level, and force structure. All of the information used in the study is preserved in the ARCHITECT environment, which is dynamic and allows for on-the-fly analysis. The assumptions used were consistent, which was assured through the use of single file documenting all inputs, which was shared across all models. Some of the other criteria are not as obviously seen through this example, such as the ability to perform multiple missions

and the reduction of cognitive biases. The ability to perform across multiple missions can be easily seen by performing another notional study, but the reduction of cognitive biases can only be seen through a physiological analysis of actual analysts and decision-makers implementing the methodology. Thus, until the method is adopted on real CBA studies, it would be difficult to test these claims. As such, the logical arguments presented in the development of the methodology are the best proof available at the present time.

The SEAD mission demonstrated the need for CBAs to explore a broad range of operational and materiel solutions. It was not only the inclusion of a given set of systems in performing the mission, but also the ways in which those systems were used that drove the performance results. Despite the notional nature of the data, the results intuitively make sense for a pre-planned SEAD mission: that stand-off assets are preferred over those that have to be in harm's way, that the ability to overwhelm the opponent by numbers is key in suppressing air defenses, that tasks such as battle damage assessment and discriminating decoys are best done by assets that include humans and are able to fly low to the ground (the AH-64 in this example), and that while increasing net-centricity does result in some increased performance for SEAD, it is not the largest driver by far. It is expected that these observations may not be true on another mission or in cases where SEAD is not pre-planned. Intuitive thought these results may seem, the ability to show the quantitative impact of each of these is critical to performing CBA, and represents a improvement over previous CBAs as described by the literature.

It was also seen that assumptions have a big impact and need to be validated and consistent throughout study. For example, the use of F-18s was eliminated early on based on inadequate performance as compared to other alternatives. However, it was later seen that a driver for this was the assumption that only a small number (less than 6) would be used to conduct the mission. When this assumption was varied

to demonstrate the impact of the assumption, it was seen that while 6 F-18s was ineffective in the assumed scenario, 9 F-18s actually performed better than most of the alternatives considered in the final downselection. Thus, all assumptions used in the study should be well documented and validated, and should never be made for simplicity. While this was already recognized as a key factor in CBA, it is shown here to be critical to having confidence in the results.

CHAPTER VI

CONCLUDING REMARKS

6.1 Summary of Findings

The research conducted as part of this thesis developed a capability-based systems engineering methodology for the early phases of design and acquisition. In particular, this method targets the CBA phase of the acquisition process, and works to improve the overall quality of information available for conducting CBA by implementing more rigorous metrics derivation and gap analysis, providing a comprehensive process for developing architectural alternatives, providing more a more quantitative and complete analysis process, and including sound decision support principles. In order to meet the overall Research Objective, seven Research Questions were presented in Table 4. In response to question 0, 15 criteria were developed based on literature from cognitive psychology, management science, SoSE, and defense acquisition. These criteria represent attributes of acquisition methodologies that have been found to improve acquisition outcomes in each respective field. These criteria formed the basis against which potential enablers for the other research questions were judged. In response to Research Question 1, which focused on the overall process model, a vee process model was selected as the most appropriate for the ARCHITECT methodology because it was well suited to incremental updates to an SoS, and has been suggested previously in the literature as appropriate for this application. The second Research Question, which addressed the metrics derivation step of the process, was answered using a combination of ROSETTA and the PSM technique to derive the set of metrics to be used in the remainder of the method. These techniques were

selected for their repeatability and traceability in particular. The third research question, covering the gap analysis step of the methodology, was again addressed using ROSETTA, this time paired with a gap analysis methodology taken from the ecology community. These methods were selected to provide a rigorous and repeatable platform for gap analysis that was able to account for uncertainty. The alternative identification step was addressed in the fourth research question, which was answered using a combination of TESSA (for describing the alternative space) and RAAM and ARCNET (for generating the architecture alternatives). TESSA was developed as a part of this research, which RAAM and ARCNET are existing tools that were leveraged for this application. These techniques allow decision-makers to clearly describe the alternative space across a broad spectrum of alternative types while automating the actual generation of the alternatives to decrease generation time and allow for the consideration of a more complete alternative space. The fifth research question dealt with the alternative evaluation. This was addressed through the development of a hierarchical evaluation and downselection approach, which begins with the application of high-level, rapid analysis tools to pare down the alternative space to regions of interest. As the evaluation and downselections proceed, the tools used become higher in fidelity but have a longer execution time. At each, stage, confidence in the modeling results increase, and the alternative space is further pruned. This approach was selected because it allowed all alternatives to be evaluated using common models with consistent assumptions, and allowed a much wider range of alternatives to be considered quantitatively in the study. Finally, the sixth research question, dealing with decision support, was addressed through the recommended application of sound decision-support principles and the recommended application of visual analytics to reduce decision-maker biases and maximize the amount and clarity of information available to the decision-maker. A summary of how each of the research questions was addressed is provided in Table 16, where the numbers refer the original number

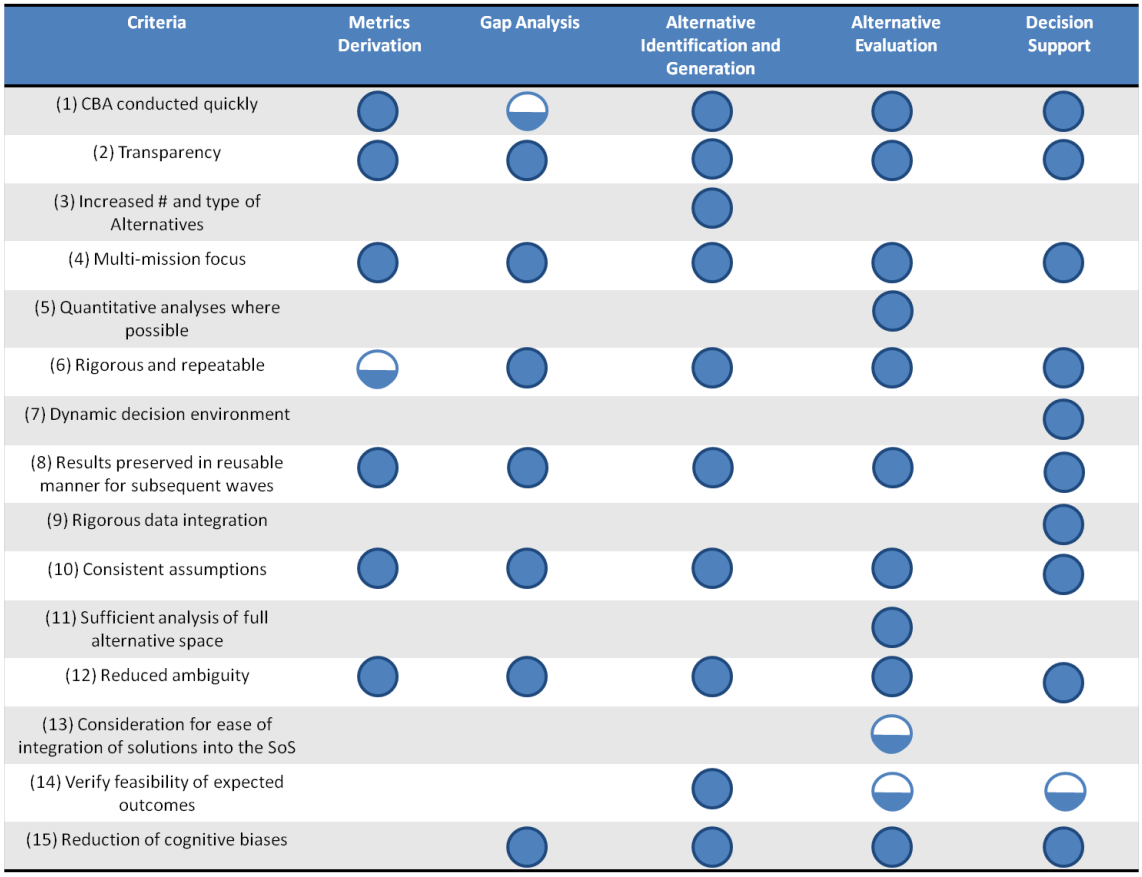


Figure 111: Assessment of the ARCHITECT Methodology Against the CBA Criteria

of the research question in Table 4.

The selection of techniques for these steps was done by mapping the set of criteria as to what is needed for a acquisition against the steps in the ARCHITECT method, and choosing the technique which best met this criteria. In the absence of a technique to meet these criteria, a new technique was developed that was better able to meet the criteria. As techniques for each step were chosen and developed, an assessment of each of these techniques against the relevant criteria was performed. The compilation of these assessments is presented in Figure 111.

Some of the criteria presented in the table can be assessed in a more quantitative fashion. In particular, two of the original motivating criteria can be assessed more

Table 16: Summary of How Research Questions Were Addressed

Number	Question
(0)	A list of criteria for success were developed by studying related fields and were presented in Table 6.
(1)	(1.1) The vee model working inside the wave model was found to be most appropriate for incremental SoS updates.
	(1.2) An N-squared diagram was used to arrange the steps.
	(1.3) The inputs and outputs for each step were identified, as was the source of each input. The inputs and outputs are summarized in Figure 44.
(2)	(2.1) The ROSETTA framework combined with the PSM technique is recommended for metrics derivation.
	(2.2) Use of the INCOSE or other similar criteria are recommended to test for metric goodness.
(3)	(3.1) The ROSETTA framework combined with the ecology approach is recommended for gap analysis.
	(3.2) A ranking that combines a range of size and criticality values to establish a range of potential gap scores was proposed to explore gap rankings.
(4)	(4.1) TESSA, RAAM, and ARCNET are combined to identify and develop the alternative space.
	(4.2) The application of relational transforms and hierarchical filtering in RAAM allow architecturally infeasible alternatives to be removed from consideration.
(5)	(5.1) A hierarchical modeling and downselection approach was developed, in which modeling breadth and depth are traded as the user moves through the hierarchy was developed
	(5.2) A subset of DoDAF products is combined with a set of metadata to support inputs to the modeling process.
	(5.3) A set of candidate modeling tools for early-phase acquisition was identified (including RAAM, ARCNET, DES, System Dynamics, Markov Chains, and Mathematical Graphs). For each, the required DoDAF products and additional supporting information was identified, as well as the types of metrics that could be captured with that model.
	(5.4) A single data integration and visualization environment is suggested to integrate modeling results.
	(5.5) In general, the OV-1, OV-2, OV-3, OV-4, OV-5b, SV-1, SV-2, OV-4 are recommended to support early-phase executable architecting. A mapping between the candidate modeling techniques and DoDAF products is shown in Table 9
(6)	For decision-support it is recommended that best practices from decision theory be applied, and that visual analytics be used in the development of a decision support environment.

rigorously: the time to conduct a CBA and the increased number and type of alternatives. With respect to the time required to conduct a CBA, the original goal was set that to perform a CBA should take less than a year. Assuming that these timeline for the CBA would be fixed by someone above the manager responsible for the CBA, the methodology must be able to be completed within this time frame. For the SEAD mission presented in this work, it took approximately three months to perform the ARCHITECT method for this example problem. However, as methodology development was being doing simultaneously with this implementation, and as the SEAD mission is only a single example that has in some ways been simplified, a simple model has been developed to better estimate the expected required time to complete the ARCHITECT method. To this, estimates were made regarding the time needed to do each step for single mission study, roughly equivalent in size to the SEAD study presented in the preceding chapter. These times were based on experience with the SEAD mission, and were, to be conservative, treated as a lower bound to the amount of time required to perform each step. Then, an expected upper bound was estimated for how long each step would take if three missions of roughly the size of the SEAD mission were to be considered. Within those bounds, a beta distribution with the alpha and beta parameters set to two was assigned to each of the steps. A beta distribution on the range zero to one with alpha and beta parameters of two is depicted in Figure 112 for reference. Next, since it possible that iteration is necessary between some of the steps, a probability of returning to the previous step was placed on each step. Decision support was considered an independent step for modeling simplicity, but the times were estimated to represent all of the decision making that must take place throughout the execution of the method. The inputs used are shown in Table 17. The control graph for this model is shown in Figure 113. The source code for the model can be found in C.

A 10,000 case Monte Carlo was then run using this model. The PDF and CDF

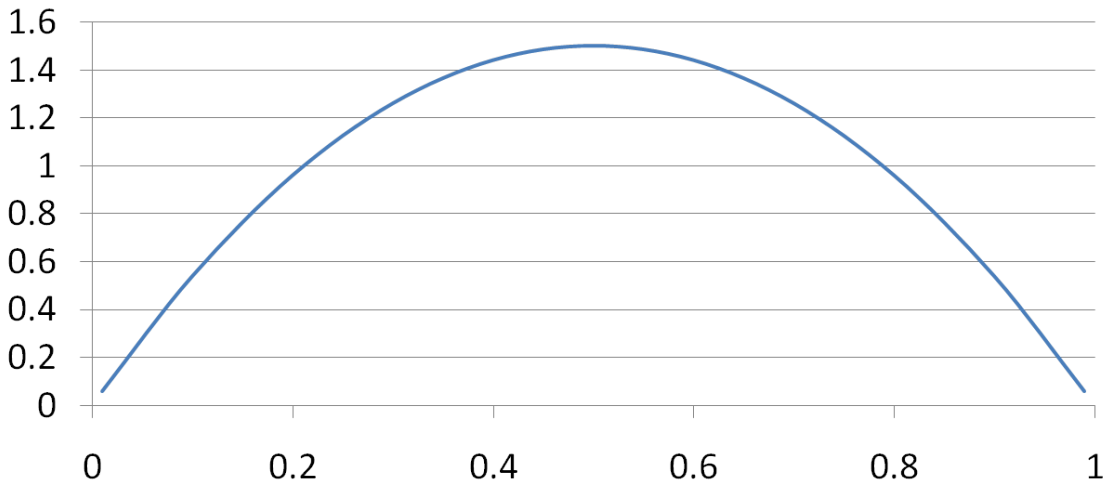


Figure 112: Beta Distribution with $\alpha = 2, \beta = 2$, and range 0 to 1

Table 17: Assessment of Times to Complete Each Step of ARCHITECT

	Problem Definition	Metrics Derivation	Gap Analysis	Alt ID	Alt Eval	Decide
Min Weeks	1	2	3	3	8	3
Max Weeks	3	4	6	6	16	6
P(iterate)	0	0	0.2	0.1	0.1	0

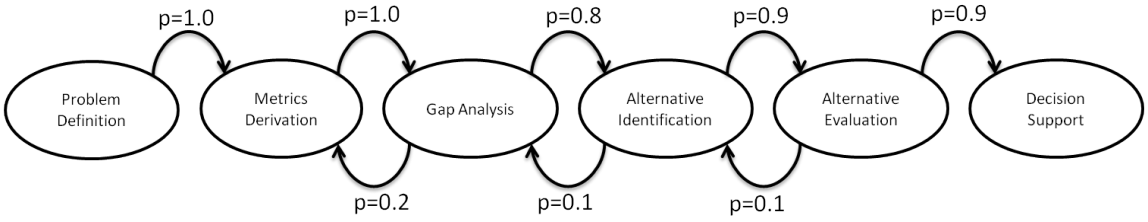


Figure 113: Control Graph for ARCHITECT Time Model

of the results are shown in Figure 114. Based on this model, it is expected that over 90 percent of CBAs could be completed in less than a year using the ARCHITECT method, and that more than half could be completed in less than 8 months. These are conservative estimates, and thus it is concluded that ARCHITECT sufficiently meets the criteria that the time of a CBA should be reduced to less than a year. However, simply improving the timeline is not alone sufficient. It would be equally easy to decrease the time required to perform a CBA by sacrificing the quality and fidelity of the study. However, it should be noted that by meeting the other criteria presented here, such as the inclusion of more quantitative analysis, the use of increased rigor, and the consistency of assumptions, the ARCHITECT methodology is able to meet the reduced timeline while still providing more quantitative, rigorous, and traceable analyses.

The second criteria which can be quantitatively measured is that the ARCHITECT methodology examines a larger number and type of alternatives than are typically considered in CBA studies. As it was observed previously that most CBAs only examine materiel solutions and consider less than 50 alternatives (and often as few as 2), it is clear that the ARCHITECT methodology is able to examine far more. The SEAD study quantitatively evaluated over 700,000,000, and these included operational, materiel, organizational, and network alternatives.

With respect to the other criteria, a qualitative discussion can be presented demonstrating that these criteria are met by the ARCHITECT methodology. First, because the ARCHITECT method stores and shows all data, uses quantitative modeling wherever possible, and documents all assumptions, it has the property of transparency. It is able, as was discussed in the methodology development, to handle multiple missions. Leveraging the RAAM framework developed by Iacobucci [87] allows for quantitative analysis to be performed where it was previously infeasible to do so because of the computational intensity required by traditional modeling paradigms. The removal of

Quantiles

100.0%	maximum	2.0666
99.5%		1.4087
97.5%		1.15048
90.0%		0.91969
75.0%	quartile	0.75026
50.0%	median	0.62124
25.0%	quartile	0.57281
10.0%		0.54158
2.5%		0.51236
0.5%		0.49023
0.0%	minimum	0.44938

Moments

Mean	0.6847316
Std Dev	0.1719242
Std Err Mean	0.0017192
Upper 95% Mean	0.6881016
Lower 95% Mean	0.6813615
N	10000

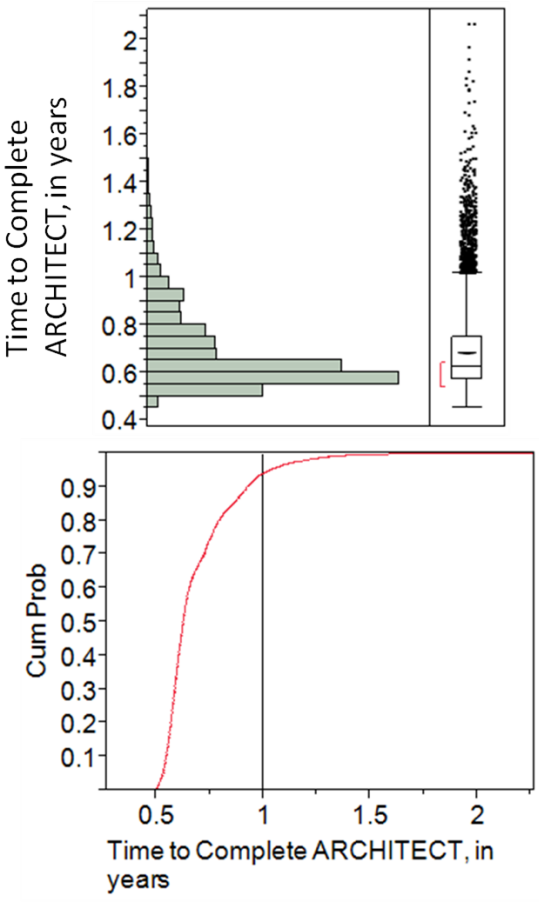


Figure 114: PDF and CDF for estimated time to complete ARCHITECT

ad-hoc methods from the CBA process increases the rigor, and the documentation of all assumptions and modeling results increases the repeatability. ARCHITECT leverages principles of visual analytics to culminate in a dynamic decision framework. Because the results are preserved, as are the models, reuse for future waves is possible. Use of a single, integrated environment to develop the baseline, the metrics, the performance thresholds, and the modeling inputs causes consistent assumptions. Furthermore, these consistent assumptions allows and integrated framework increase the rigor of the data integration by ensuring that all data was developed was developed from the same initial starting point and is all brought together into a common location and in a common format. The increase in quantitative analysis, the use of consistent assumptions, the increased transparency, and the rigorous data integration all contribute to a reduced ambiguity. Also contributing to this is the ability to include uncertainty in the quantitative analyses. The consideration for the ease of integration and the verification of the feasibility of expected outcomes are not yet fully addressed. The current ARCHITECT method has begun to think about how to address these issues, but in the future needs to be expanded to treat them more rigorously. Finally, as has been discussed throughout the methodology development, significant effort has been made to reduce cognitive biases in each step of the process.

The SEAD study presented demonstrated the overall plausibility of the ARCHITECT methodology, showing that the techniques selected for each step can not only be used individually to enable each step of the methodology, but can also be used together in a complementary way to execute the methodology as a whole, gathering or creating the needed information at each step to support the next one. The SEAD study also demonstrated the ability of the methodology to perform high-level architectural trades that allow for generalizations regarding how certain characteristics of an architecture drive the overall performance. These generalizations included generalizations about which systems were more effective, which system-to-task mappings

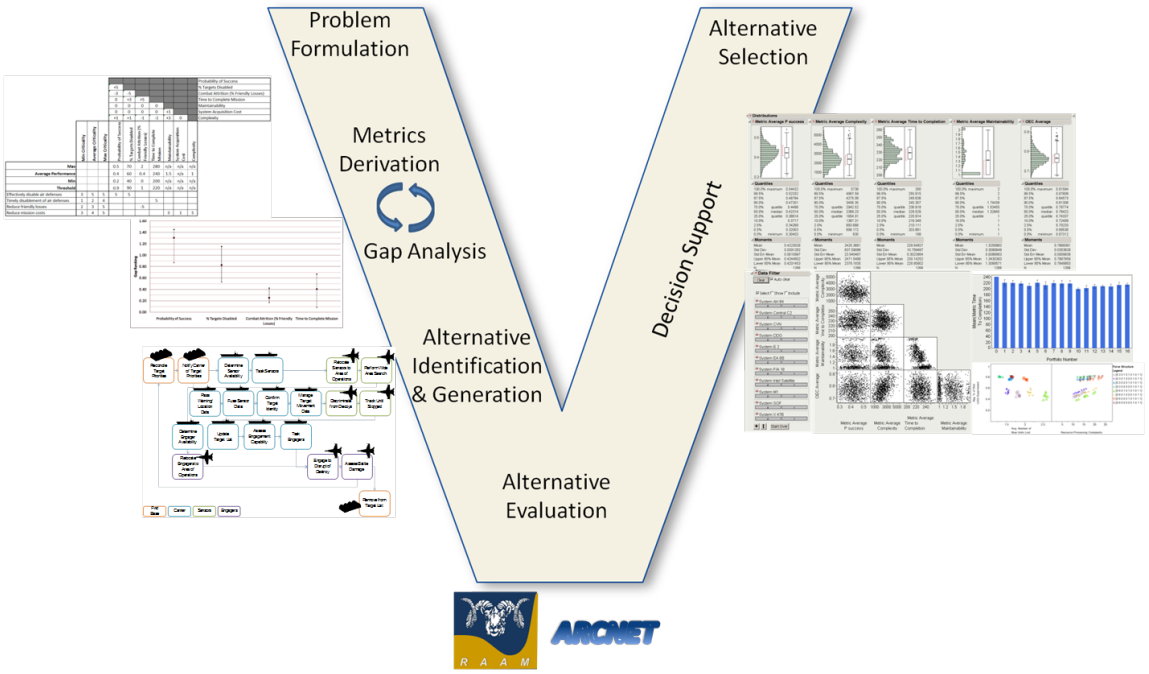


Figure 115: Visual Summary of the application of ARCHITECT to SEAD

had the greatest impact on success, and the trade between using force-by-numbers versus smaller forces with networked effects. These generalizations could be mapped to quantitative performance increases. The assumptions, baseline, models, data, and architecture alternatives are all stored in a reusable framework for future use. A visual summary of the application of the ARCHITECT methodology to the SEAD study is shown in Figure 115

The ARCHITECT methodology developed as the result of this research was designed to enable improved CBA in a military system of systems context as part of the JCIDS process by designing a methodology which met a list of criteria that have been shown to improve acquisition outcomes in both the DoD and other walks of life. It is specifically designed to help with the identification of needs, identification of potential and varied alternatives to meet those needs, and to help perform a set of initial downselections so that decision makers can make a quantifiable and traceable decision at Milestone A as to whether or not a materiel solution is needed. This will allow decision-makers to identify and then prune down the large SoS alternative space

available in early-phase decision-making. To complete the analysis for a Milestone A decision, more detailed modeling for the final alternative set would be required to study the integration effects for the SoS and refine the performance estimates from the initial ARCHITECT analysis. Although it is expected that the ARCHITECT methodology will apply to early-phase systems engineering for a broader class of problems, this has not been tested.

6.2 Summary of Contributions

This work has resulted in several contributions to the fields of SoSE and CBA. First, a cross-domain literature search has combined information from CBA literature, SoSE literature, historical lessons learned from corporate acquisitions, and cognitive psychology studies on strategic decision-making to create a comprehensive list of criteria which should be used to guide the creation of and assess the goodness of CBA methodologies. This list of criteria has further been decomposed to show which criteria apply to which phases of the process. A CBA methodology has been developed which meets these criteria, and is available in the public domain. This methodology leverages concepts from systems engineering, quality engineering, architecture-based design and analysis, and executable architecting. In conjunction with the development of the ARCHITECT methodology, the ROSETTA framework was developed as a general framework for applying relational-oriented systems engineering. As part of the development of the ARCHITECT methodology, the ROSETTA framework has been extended to the SoS space, and has been applied in several new ways, including for the purposes of metrics derivation and gap analysis. Furthermore, an equivalent to morphological analysis for SoS has been developed to fully develop and describe the SoS alternative space in a structured manner, and to eliminate those which are infeasible from consideration. A hierarchical approach to evaluating and comparing the full, large alternative space for an SoS using quantitative modeling has also been

developed.

6.3 Future Research Areas

While the ARCHITECT methodology creates a solid initial foundation for conducting CBA, there are many areas of future work which could further improve the ARCHITECT methodology and could help to extend the applicability. There is room for improvement in the area of uncertainty, and a more formal treatment of uncertainty would act to increase confidence in the results. Additionally, guidance on the verification and validation of models used in support of the ARCHITECT method would be beneficial to users and would also help to increase the confidence in the results of the ARCHITECT method. Another area of future work is to further explore how the input parameters for ARCHITECT can be estimated. Several ideas for the estimation are proposed in the methodology, but a formal comparison of these approaches as well as other ideas on how he estimates of the required inputs can be obtained with increasing confidence would be a beneficial next step. Furthermore, exploring more of the ideas used by strategic decision-makers in corporate acquisitions may hold much promise for improving the ARCHITECT methodology. In particular, strategic decision-making research has heavily researched when in the design and development process down selections should be made, and the application of the research to explore the timeline associated with the ARCHITECT method would be of interest. While some steps have been taken to identify and reduce decision-maker biases, further research in this area could help to uncover and address even more potential biases. A more thorough exploration of existing decision-support techniques and principles would benefit the ARCHITECT method. A better treatment of both the consideration of the ease of integration of solutions into the existing SoS and and exploration of verification are both also areas which would benefit greatly from further research.

A more detailed study of interoperability, the complexities it adds to the decision-space, and how to better capture interoperability trades during early-phase systems engineering is of interest. A study comparing the results of each step of the methodology using qualitative and quantitative data may provide additional guidance as to when each is necessary. This could be aided by further implementation of the ROSETTA framework within the ARCHITECT methodology as the data structure to capture and store both qualitative and quantitative information. Implementation of ROSETTA in this way would add structure to the methodology. Finally, only time and a broader range of applications can fully demonstrate the utility of the methodology and its ability to apply for multi-mission analysis, and this is the most obvious and necessary next step in the research.

APPENDIX A

RELEVANT DODAF MODEL OVERVIEW

The information contained in this section is based on [46] and [50]. This is not intended to be a summary of DoDAF, but rather a brief overview of some the relevant DoDAF models.

The AV contains information that is applicable to all other views. They provide the overarching concept and scope of the architecture. The AV has two products, the AV-1 and AV-2. The AV-1, Overview and Summary Information, contains high-level information about the architecture, including the scope and purpose, the intended users, and the intended operational environment. IT also includes doctrine; tactics, techniques, and procedures (TTP); relevant goals and vision statements; concepts of operations (CONOPS); and scenarios. The AV-2, Integrated Dictionary, provides a dictionary of the terms used in the products. This helps keep terminology consistent and allows those from different disciplines and domains to communicate effectively with a set of common terminology.

The Operational View (OV) has 7 products, but the OV-6 actually contains three parts, resulting in a total of 9 products. The operational view looks at the needs of the architecture, rather than how the needs are carried out. It is system-independent, although many users of DoDAF have found it helpful to overlay system information (found in the Systems and Services View) on top of operational products to paint a more complete or easier to understand picture. The OV describes important nodes, information exchanges, and activities independent of the systems. The OV-1, High Level Operational Concept Graphic, provides a simple, high-level description of the architecture, including basic components and their overall role in performing the

duties of the architecture. The OV-1 is essentially the executive summary of the architecture, presented in a graphical format. The OV-1 briefly captures the mission of the architecture, the basic operational concept, and important high-level interactions with the environment and between components. There is no recommended format for this product, as it is primarily used as a communication tool. The OV-1 is useful in the early stages of problem definition to help stakeholders 'get on the same page' and in scenario definition and scoping.

The OV-2, Operational Node Connectivity Description, largely depicts operational nodes and the needlines between those nodes. An operational node is defined as "a node that performs a role or a mission", where a node is "A representation of an element of architecture that produces, consumes, or processes data." This implies that nodes are able to independently accomplish some function that is needed to perform the larger mission. A needline is defined as "a requirement that is the logical expression of the need to transfer information among nodes." Put another way, a needline denotes that a given node requires information that is generated in another node, and the needline connects the node in which the information originates to the node in which the information is consumed. Typically, needlines are represented with directional arrows to indicate the direction of information flow, and needline identifiers that allow specific information exchanges to be associated with their appropriate needline (see OV-3). Often operational activities performed by a node are listed next to the node in this product (see OV-5). One of the most important features of the OV-2 is that it tracks only the need for information, and not the path that is taken for the information to get there. This means that it is particularly useful when brainstorming potential operational alternatives for performing a mission because it is not biased by the way things are currently done. It simply documents which information must travel where for mission success.

The OV-3, Operational Information Exchange Matrix, details the information

exchanges that occur across the needlines of the OV-2. An information exchange is the exchange of a collection of information elements, which are defined as "Information that is passed from one operational node to another. Associated with an information element are such performance attributes as timeliness, quality, and quantity values." For each needline, all information exchanges across that needline are documented, along with relevant details about each information exchange, including information such as who exchanges the information, what information elements are exchanged, why the information is needed, the classification level of the information, and any other relevant details. This product, combined with OV-2, creates a complete list of information needs and necessary flow, and is thus very helpful in early stages of design and problem formulation.

The OV-4, Organizational Relationships Chart, describes the relationships between organizations and agencies within the architecture. The OV-4 primarily describes the role of people in the architecture, and can be used to depict command or management structure. It can describe who has authority in which situation and identify governing roles. It can also describe cooperative relationships in which one party has no authority over the other. The OV-4 is very useful in understanding the roles of stakeholders and the authority of stakeholders. It can help identify which pieces of the architecture are governed by which organization, and who claims responsibility for which elements. It can also help identify situations in which authority is duplicated or unclear. When looking at potential changes to the architecture, or when designing an architecture, it is important to understand which pieces of the architecture fall into whose jurisdiction so that the proper organizations can be consulted. If a specific organization is looking at making changes, it is important to understand what areas of the architecture that organization has the authority to change so that only feasible solutions can be considered.

The OV-5, Operational Activity Model, describes the operational activities that

need to be conducted in order to achieve a capability, and the order in which those activities must be performed. An operational activity is defined as "an action performed in conducting the business of an enterprise". In general capabilities are defined by one or more sequences of activities, which are called operational threads. This product is typically depicted as a hierarchy or a flow chart. The OV-5 is one of the most utilized products in DoDAF. This is likely because the OV-5 provides a fundamental sketch of how a capability is actually achieved in practice, and breaks down the capability into the steps that must be completed to claim success. It is very useful in understanding the capability, and in identifying gaps. It is often simpler to identify a gap by an action that cannot be completed fast enough, well enough, cost-effectively enough, or at all than to speculate about the gaps with no basis for judgment.

The OV-6 and SV-10 series of DoDAF products are called 'dynamic' products. This does not mean they are dynamic in the sense that they are reconfigurable or executable, but that they contain information about how events occur in the time domain. More specifically, these products contain information about the timing and sequencing of events and capture the operational behavior of a business process or a mission thread. While other products examine what the nodes or systems are, for what tasks they are responsible, what information is shared, or who is responsible, the OV-6 and SV-10 products look at how activities are carried out and what behavior is required to enable successful use of the architecture.

The OV-6 is one of the products that most naturally lends itself to modeling. It has three parts: the OV-6a, Operational Rules Model, the OV-6b, State Transition Description, and the OV-6c, Event-Trace Description The OV-6a contains a description of the operational and business rules that constrain the operations of the architecture. The OV-6b describes the state changes in operational nodes or activities resulting from specific events. It provides the time sequencing for the OV-5. The OV-6c gives a time-sequenced description of the information exchanges that take

place between operational nodes for a given scenario or capability. This shows not only what information sharing is required between nodes, but when that information sharing must take place in order for the mission to be successful. The DoDAF documentation actually recommends the use of modeling with this product. In fact, about the set of OV-6 products in general, Volume 2 of the DoDAF documentation says "Several modeling techniques may be used to refine and extend the architecture's OV to adequately describe the dynamic behavior and timing performance characteristics of an architecture, such as LDL [Naqvi, 1989], Harel Statecharts [Harel, 1987 a, b], petri-nets [Kristensen, 1998], IDEF3 diagrams [IDEF3, 1995], and UML statechart and sequence diagrams [OMG, 2001]."

The Systems and Services View (SV) depicts the roles of the systems in fulfilling the needs of the architecture as outlined in the Operational View. The SV focuses on what systems reside at which operational nodes, how information exchanges are technically implemented, what function each system performs, and how these functions map to the activities defined in the OV. The SV has 11 products, although several of these have multiple parts. The SV-1, Systems/Services Interface Description, is the systems view counterpart to the OV-2. It identifies the system nodes and systems that support the operational nodes depicted in the OV-2, and shows the interfaces between the systems and the system nodes. This view links the operational nodes to systems and the needlines to system interfaces. However, it does not depict how the interfaces are physically implemented. This is done in the SV-2. In the SV-1, some interfaces are designated as key interfaces. A key interface is defined as an interface where one or more of the following criteria are met:

- The interface spans organizational boundaries (may be across instances of the same system, but utilized by different organizations).
- The interface is mission critical.

- The interface is difficult or complex to manage.
- There are capability, interoperability, or efficiency issues associated with the interface.

Identifying key interfaces is important because these interfaces have a high risk of failure, a higher consequence of failure, or both. Additionally, these interfaces may be locations where cross-organization or cross-agency collaboration is required, thus placing additional constraints on the interfaces during design and development.

The SV-2, Systems/Services Communications Description, includes information about the communications systems, communications links, and communications networks. Specifically, the SV-2 focuses on automated communications interfaces and the physical implementation of those interfaces, including communications systems, multiple communications links, communications networks, routers, and gateways. This means that the SV-2 can show communications paths and cycles, as well as the specific technical data associated with the physical implementation of those paths and cycles. For the purpose of modeling, the SV-1 and SV-2 can be depicted together, as a majority of models will require information about the systems and interface details.

The SV-3, Systems-Systems, Services-Systems, Services-Services Matrices, provide detailed information regarding the interfaces portrayed in the SV-1. These relationships are described using a matrix format. The types of interface details included in the SV-3 are things such as status (e.g., existing, planned, potential, deactivated), purpose (e.g., C2, intelligence, logistics), classification level, means (e.g., Secret Internet Protocol Router Network (SIPRNET)), or whether or not the interface is considered a key interface. The matrix format can be useful in supporting the rapid assessment of potential commonalities and redundancies (or, if fault-tolerance is desired, the lack of redundancies).

The SV-4, Systems/Services Functionality Description, contains two parts. The SV-4a documents system functional hierarchies and system functions, and the data

flows between them. The SV-4a is the SV counterpart to the OV-5. It depicts the functions that are performed by individual systems and their data flows to ensure that all input and output requirements are met and that the functional connectivity is complete. Sources and sinks within the data flows are also identified in this view. Because this view depicts data flows, it is useful in creating models that are based on flow analysis, such as a system dynamics model. However, the SV-4a is dependent on the specific systems being used. Changes to the SV-1/SV-2 can result in changes to the SV-4a if the function set with which activities are accomplished changes. Additionally, decisions made about how to the SV-1/SV-2 will be implemented affect the SV-4a. The SV-4b is the service-focused equivalent of the SV-4a.

The SV-5, Operational Activity To Systems Function, Activity To Systems, And Activity To Services Traceability Matrices, specifies the relationships between the operational activities and systems and systems functions. The SV-5a maps the operational activities to the system functions, depicting how specific functions performed by systems enable the performance of operational activities. The SV-5b and c map the operational activities to the systems and services that support the activities. It may be necessary to create a mapping between the system functions and systems and services that perform them, which can then be used to derive the SV-5b and c from the SV-5a. It is also possible to extend these mapping to include the specific capabilities that are supported by the operational activities, and thus show a mapping between systems and services and the capabilities which they support. It is important to realize that the systems and system functions are derived from the SV-1, SV-2, and SV-4, and the operational activities are derived from the OV-5.

APPENDIX B

ESTIMATIONS OF METRICS FOR RAAM INPUT FOR SEAD MISSION

This appendix contains documentation of the inputs to RAAM for the SEAD alternative evaluations. Note that the estimations are notional for the purpose of illustrating the method and are not intended to reflect the performance of the real systems. Tables 18 through 28 summarize the inputs used for each system, against each metric, for each task. Note that cost and risk are task-independent, and thus are listed once at the top of each table for each system. The probability of success and time are listed for each task, in turn.

Table 18: CVN Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	10000	dollars
	Risk	0.1	risk
Reconcile-Target-Priorities	P-success	0.98	prob
	Time	5	s
Determine-Sensor-Availability	P-success	0.99	prob
	Time	10	s
Task-Sensor	P-success	0.98	prob
	Time	30	s
Fuse-Sensor-Data	P-success	0.99	prob
	Time	3	s
Pass-Warning-and-Location-Data	P-success	0.97	prob
	Time	20	s
Manage-Target-Movement-Data	P-success	0.98	prob
	Time	1	s
Update-Target-List	P-success	0.99	prob
	Time	20	s
Assess-Engagement-Capability	P-success	0.99	prob
	Time	5	s
Assign-Weapon-and-Platform	P-success	0.995	prob
	Time	30	s
Remove-from-Target-List	P-success	0.96	prob
	Time	1	s

Table 19: Central C2 Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	14.5	dollars
	Risk	0.2	risk
Reconcile-Target-Priorities	P-success	0.98	prob
	Time	5	s
Determine-Sensor-Availability	P-success	0.95	prob
	Time	10	s
Task-Sensor	P-success	0.99	prob
	Time	30	s
Fuse-Sensor-Data	P-success	0.97	prob
	Time	3	s
Pass-Warning-and-Location-Data	P-success	0.99	prob
	Time	20	s
Manage-Target-Movement-Data	P-success	0.95	prob
	Time	1	s
Update-Target-List	P-success	0.98	prob
	Time	20	s
Assess-Engagement-Capability	P-success	0.98	prob
	Time	5	s
Assign-Weapon-and-Platform	P-success	0.99	prob
	Time	30	s
Remove-from-Target-List	P-success	0.95	prob
	Time	1	s

Table 20: E-2 Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	100	dollars
	Risk	0.05	risk
Fuse-Sensor-Data	P-success	0.98	prob
	Time	5	s
Pass-Warning-and-Location-Data	P-success	0.99	prob
	Time	20	s
Manage-Target-Movement-Data	P-success	0.99	prob
	Time	1	s
Track-Until-Stopped	P-success	0.98	prob
	Time	60	s
Update-Target-List	P-success	0.97	prob
	Time	20	s

Table 21: Satellite Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	3000	dollars
	Risk	0.4	risk
Wide-Area-Search	P-success	0.87	prob
	Time	3600	s
Identify	P-success	0.85	prob
	Time	600	s
Discriminate-Launch-and-Support-Systems-from-Decoys	P-success	0.8	prob
	Time	30	s
Track-Until-Stopped	P-success	0.95	prob
	Time	1	s
Battle-Damage-Assessment	P-success	0.85	prob
	Time	15	s

Table 22: X-47B Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	80	dollars
	Risk	0.8	risk
Wide-Area-Search	P-success	0.9	prob
	Time	14400	s
Identify	P-success	0.85	prob
	Time	300	s
Discriminate-Launch-and-Support-Systems-from-Decoys	P-success	0.9	prob
	Time	30	s
Track-Until-Stopped	P-success	0.95	prob
	Time	1	s
Engage-to-Destroy	P-success	0.8	prob
	Time	15	s
Engage-to-Disrupt	P-success	0.9	prob
	Time	15	s
Battle-Damage-Assessment	P-success	0.99	prob
	Time	20	s

Table 23: F/A-18 Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	68	dollars
	Risk	0.05	risk
Wide-Area-Search	P-success	0.9	prob
	Time	10800	s
Identify	P-success	0.9	prob
	Time	300	s
Discriminate-Launch-and-Support-Systems-from-Decoys	P-success	0.86	prob
	Time	30	s
Engage-to-Destroy	P-success	0.97	prob
	Time	15	s
Battle-Damage-Assessment	P-success	0.76	prob
	Time	20	s

Table 24: SOF Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	20	dollars
	Risk	0.2	risk
Identify	P-success	0.98	prob
	Time	5	s
Discriminate-Launch-and-Support-Systems-from-Decoys	P-success	0.76	prob
	Time	5	s
Track-Until-Stopped	P-success	0.98	prob
	Time	1	s
Engage-to-Destroy	P-success	0.99	prob
	Time	60	s
Engage-to-Disrupt	P-success	0.99	prob
Battle-Damage-Assessment	Time P-success	60 0.99	s prob
	Time	5	s

Table 25: AH-64 Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	20.5	dollars
	Risk	0.05	risk
Wide-Area-Search	P-success	0.8	prob
	Time	21600	s
Identify	P-success	0.79	prob
	Time	150	s
Discriminate-Launch-and-Support-Systems-from-Decoys	P-success	0.96	prob
	Time	5	s
Engage-to-Destroy	P-success	0.87	prob
	Time	10	s
Battle-Damage-Assessment	P-success	0.98	prob
	Time	10.5	s

Table 26: Mortar Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	0.25	dollars
	Risk	0.05	risk
Identify	P-success	0.999	prob
	Time	60	s
Discriminate-Launch-and-Support-Systems-from-Decoys	P-success	0.7	prob
	Time	15	s
Engage-to-Destroy	P-success	0.8	prob
	Time	30	s
Battle-Damage-Assessment	P-success	0.99	prob
	Time	60	s

Table 27: EA-6B Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	70	dollars
	Risk	0.05	risk
Wide-Area-Search	P-success	0.9	prob
	Time	14400	s
Identify	P-success	0.999	prob
	Time	300	s
Discriminate-Launch-and-Support-Systems-from-Decoys	P-success	0.8	prob
	Time	60	s
Engage-to-Disrupt	P-success	0.95	prob
	Time	15	s
Battle-Damage-Assessment	P-success	0.98	prob
	Time	15	s

Table 28: DDG Metric Estimations

Task	Metric	Avg. Value	Unit
	Cost	2000	dollars
	Risk	0.6	risk
Engage to Destroy	P-success	0.99	prob
	Time	150	s

APPENDIX C

ARCHITECT MONTE CARLO MARKOV CHAIN MODEL

The JMP® code for the model that was created to estimate the time required to complete the ARCHITECT method is provided here for completeness.

```
times={};
for(i=1,i<=10000,i++,
state=0;
time=0;
while (state < 6,
if (state==0,
timeinc=Random Beta( 2,2,1,2 );
time=time+timeinc;
state=1;);
if (state==1,
timeinc=Random Beta( 2,2,2,2 );
time=time+timeinc;
state=2;);
if (state==2,
timeinc=Random Beta( 2,2,3,3 );
time=time+timeinc;
if(Random Uniform( 0, 1 )<=.2,
state=1;,
state=3;));
```

```
if (state==3,
timeinc=Random Beta( 2,2,3,3 );
time=time+timeinc;
if(Random Uniform( 0, 1 )<=.1,
state=2;,
state=4;));
if (state==4,
timeinc=Random Beta( 2,2,8,8);
time=time+timeinc;
if(Random Uniform( 0, 1 )<=.1,
state=3;,
state=5;));
if (state==5,
timeinc=Random Beta( 2,2,3,3);
time=time+timeinc;
state=6;););
times[i]=time;);
```

APPENDIX D

SEAD ARCNET RESULTS

The results from ARCNET on alternatives 2 through 11 are contained Figures 116 through 125 in this Appendix. These can be interpreted in an analogous way to the results presented in Chapter 5. Each figure shows the results of the engagement model and the effects of varying IOL on these results.

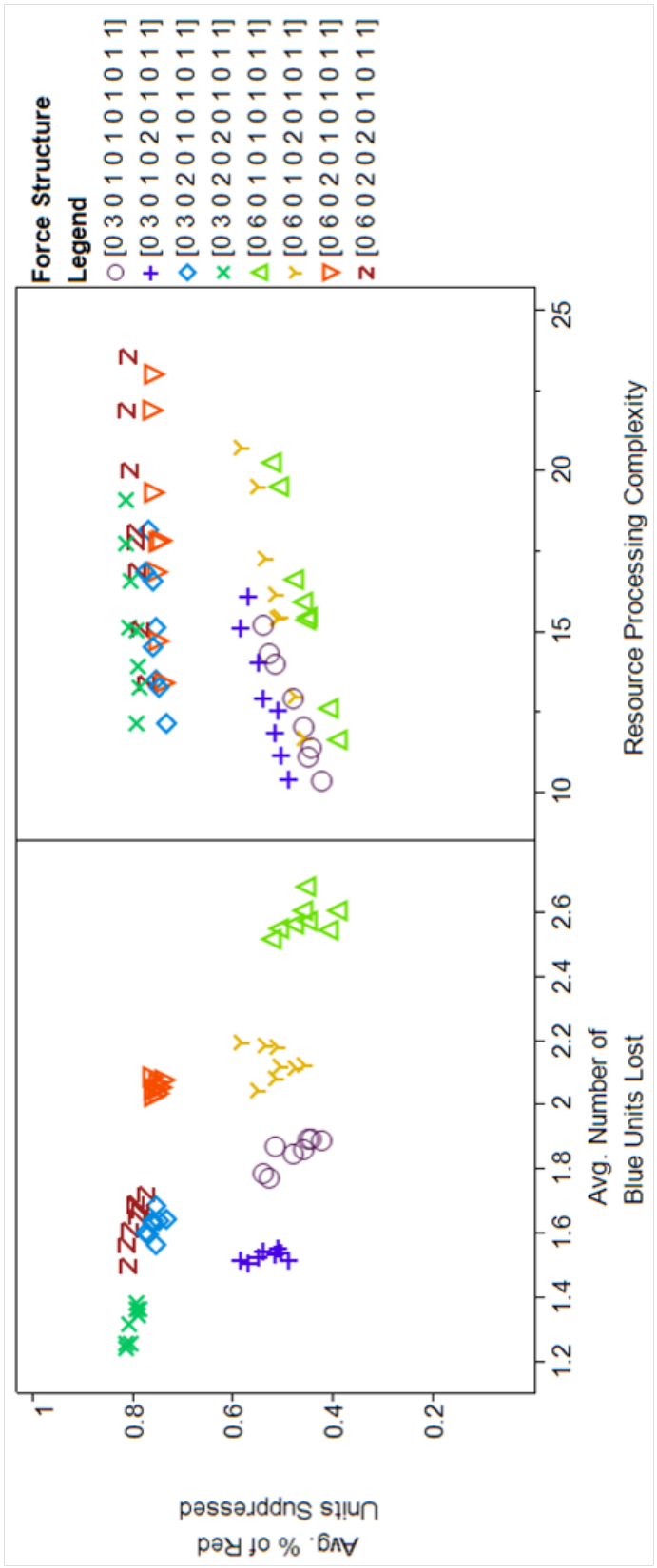


Figure 116: Alternative 2 ARCNET Results

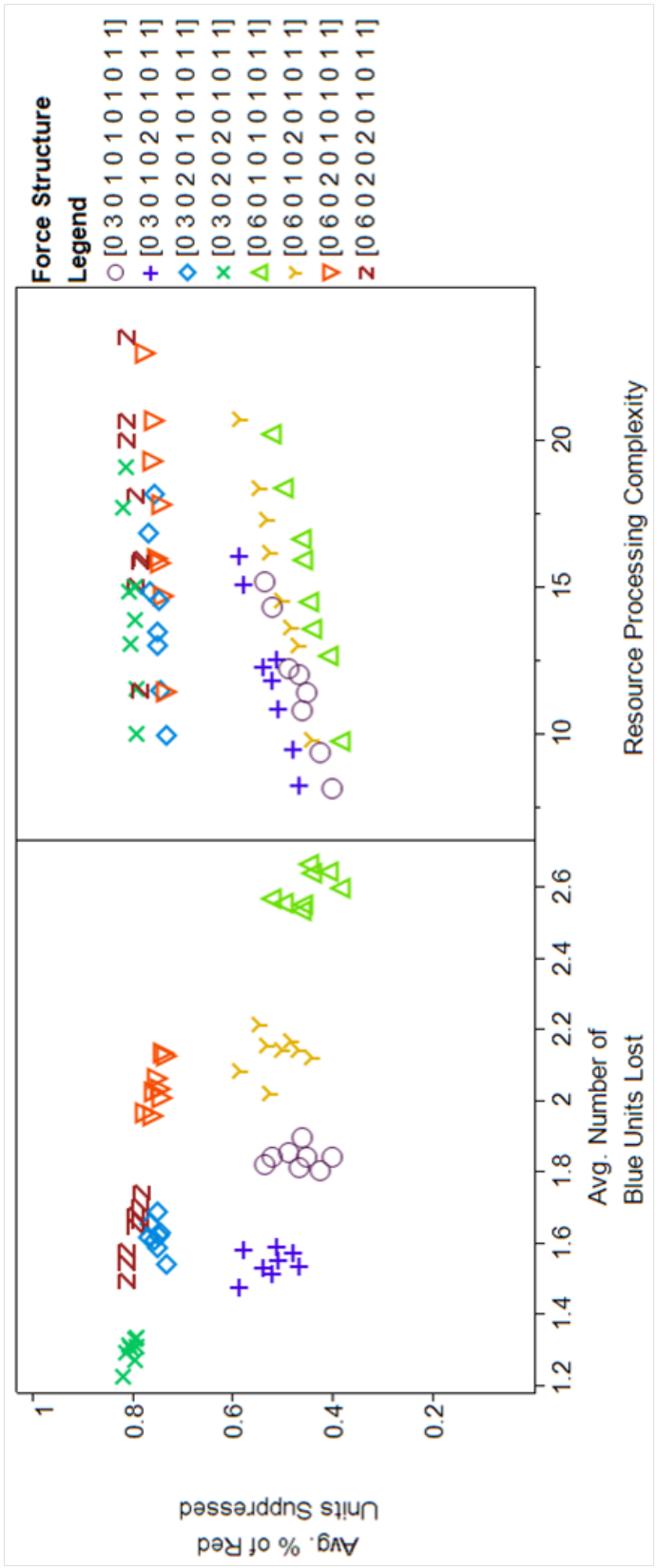


Figure 117: Alternative 3 ARCNET Results

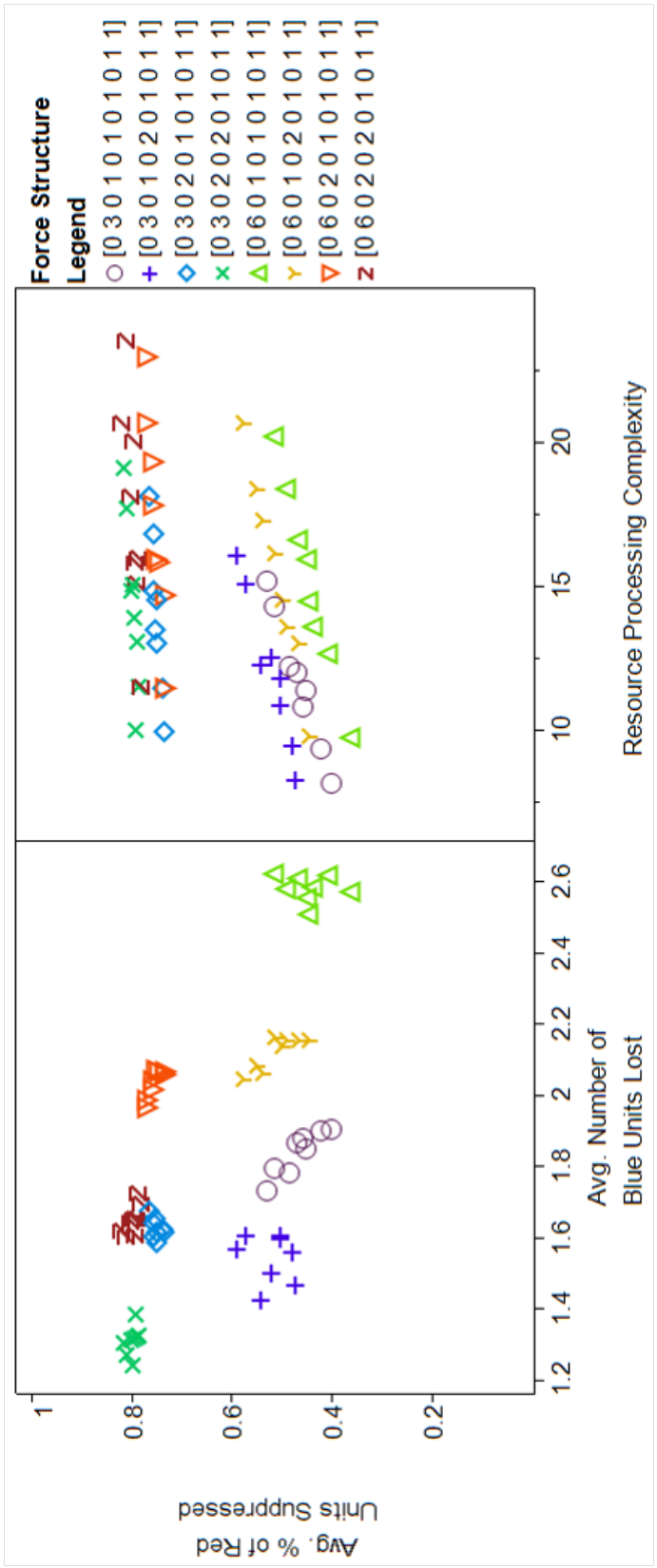


Figure 118: Alternative 4 ARCNET Results

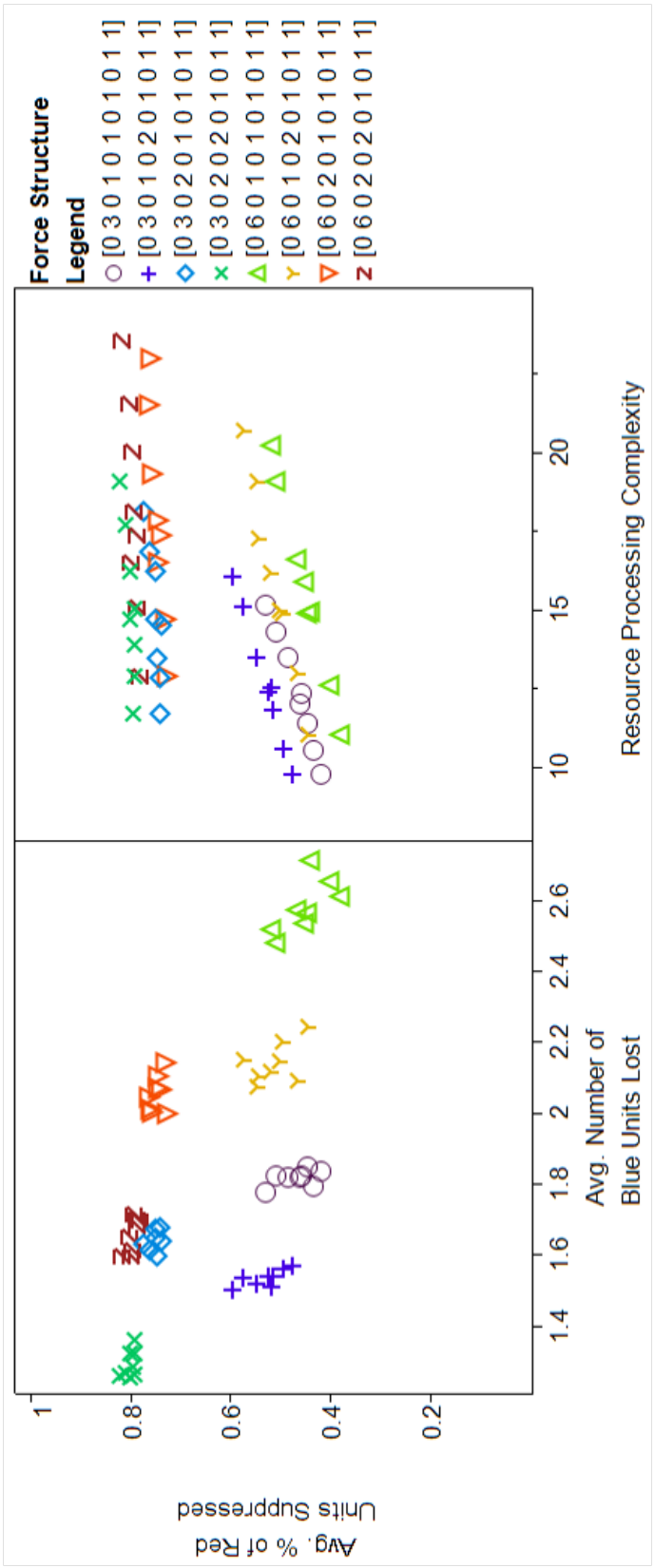


Figure 119: Alternative 5 ARCNET Results

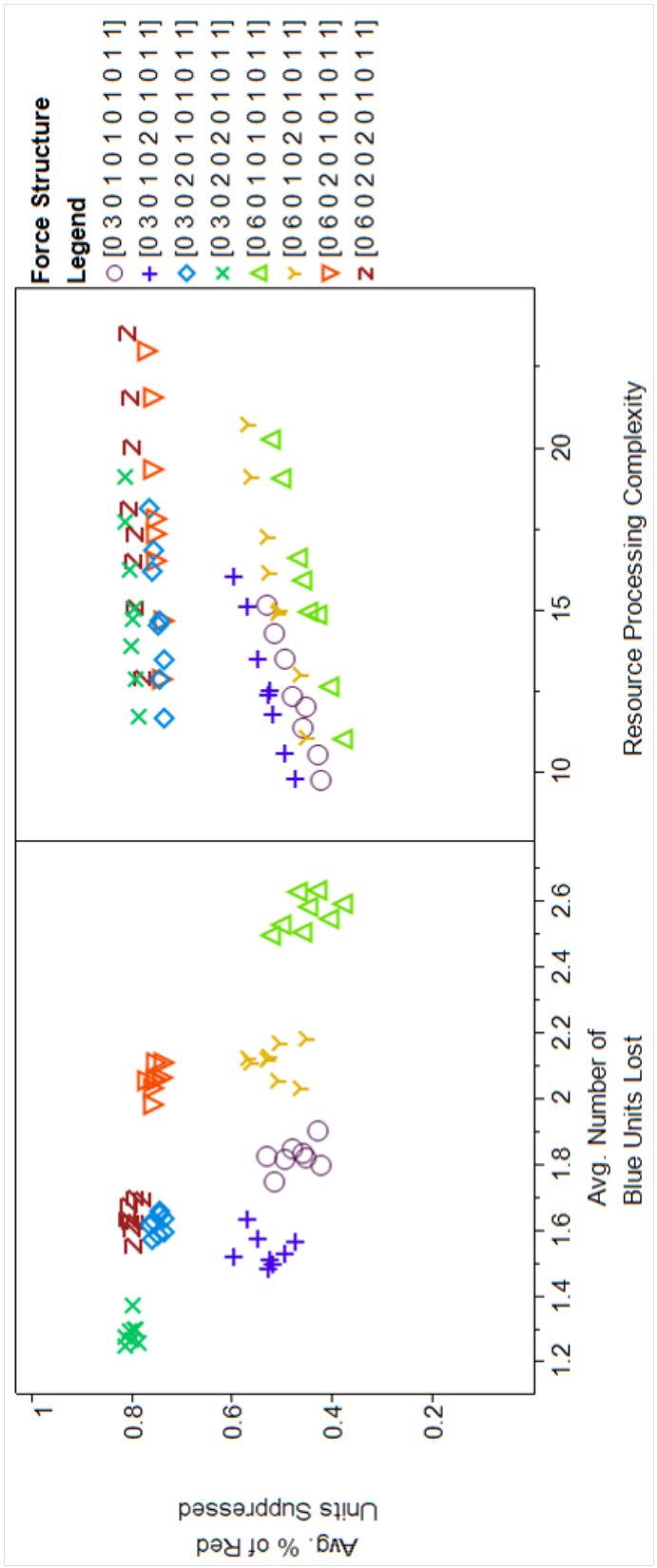


Figure 120: Alternative 6 ARCNET Results

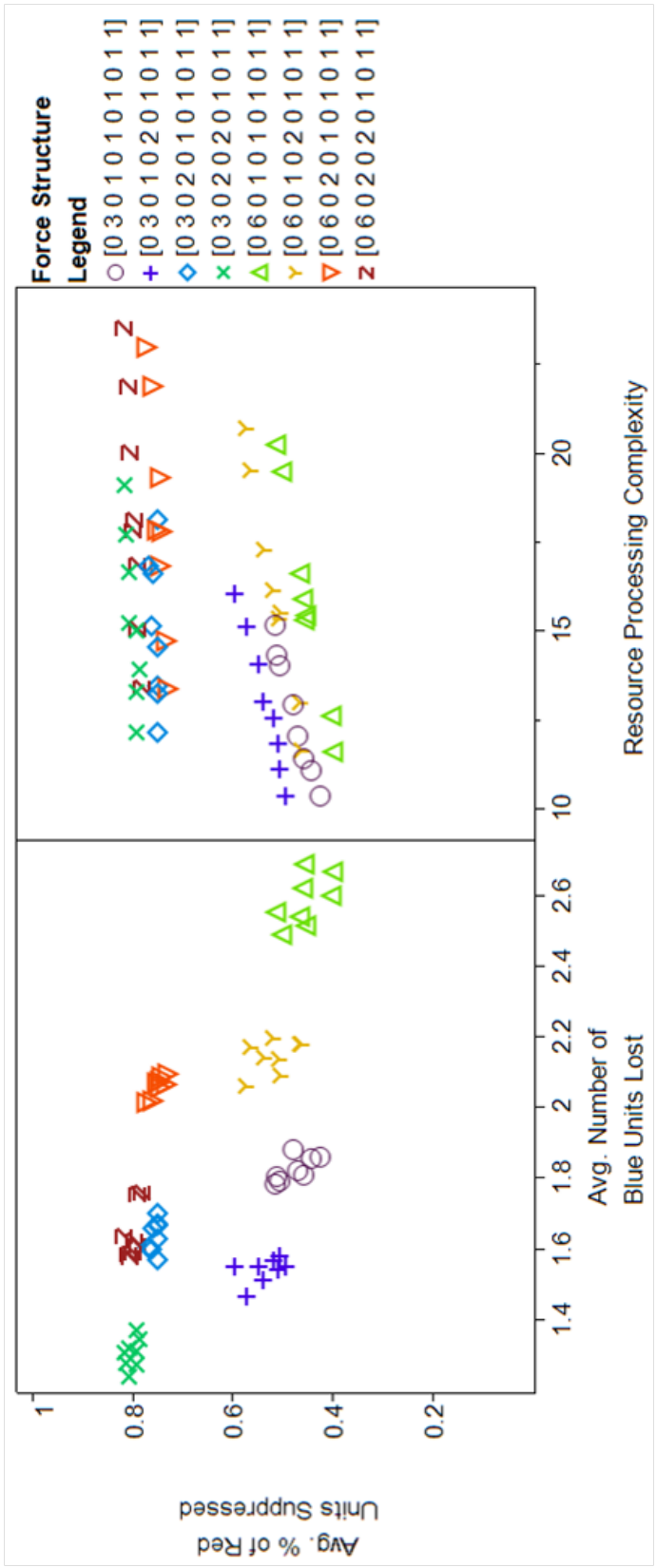


Figure 121: Alternative 7 ARCNET Results

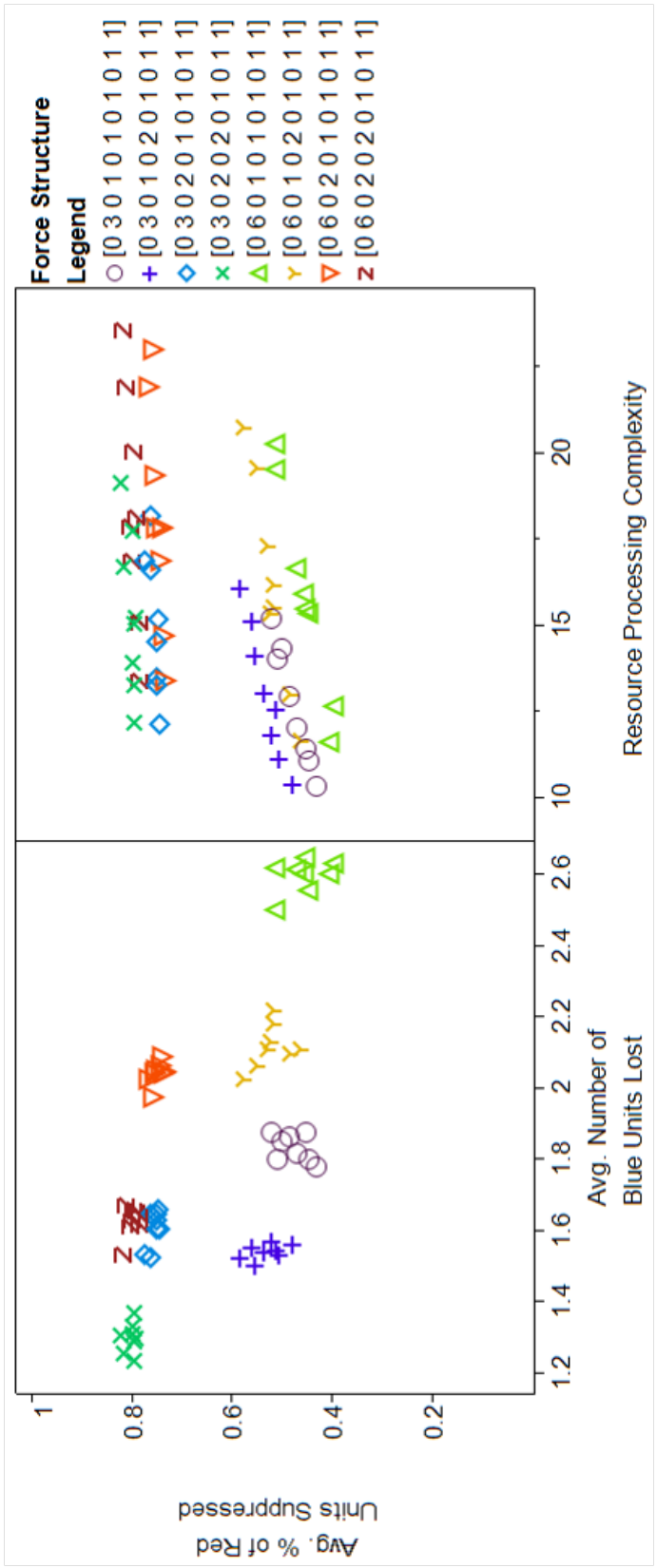


Figure 122: Alternative 8 ARCNET Results

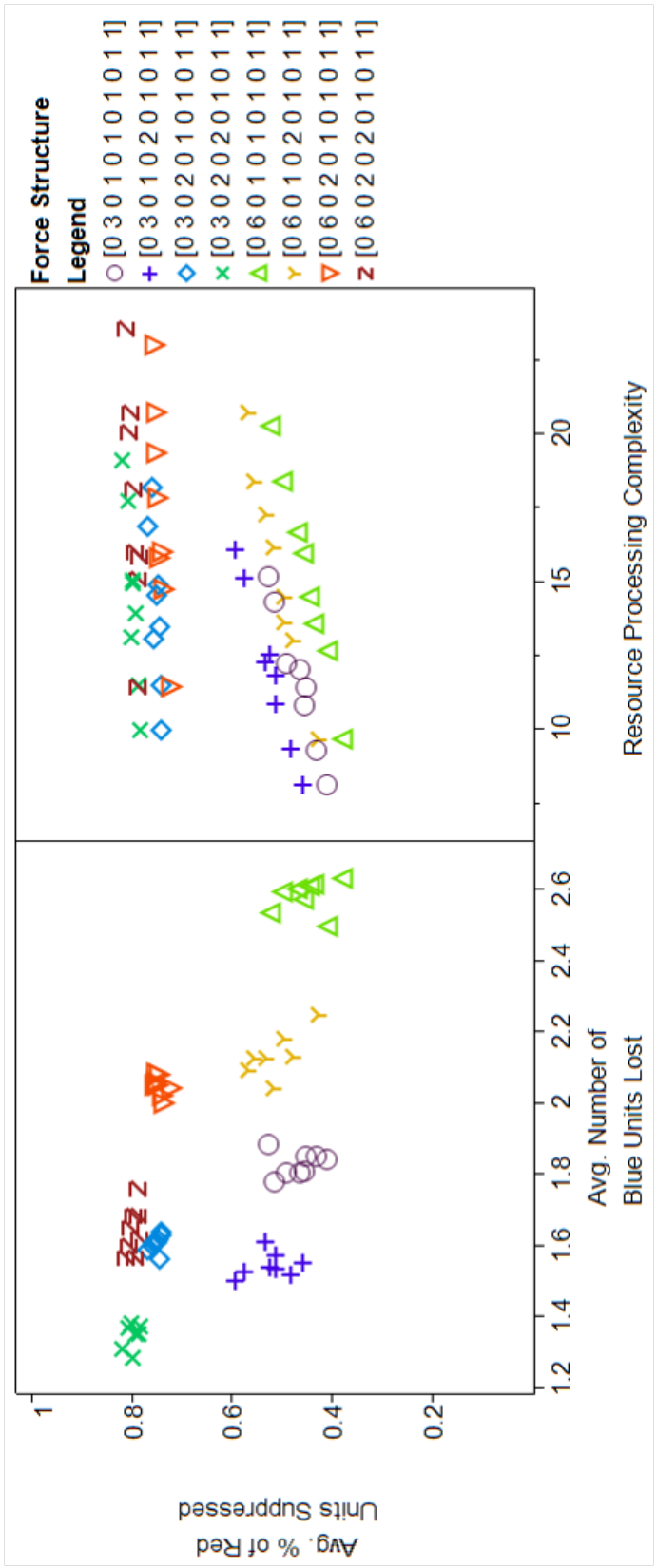


Figure 123: Alternative 9 ARCNET Results

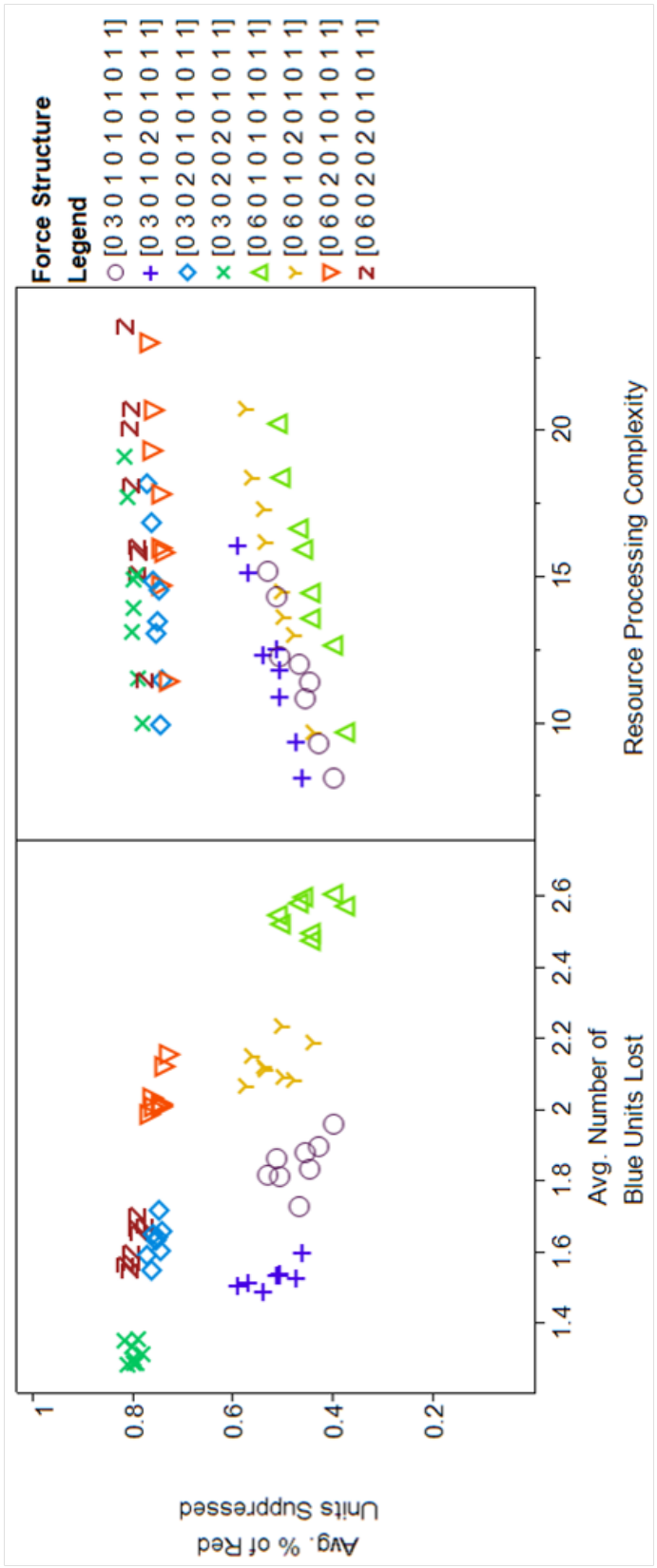


Figure 124: Alternative 10 ARCNET Results

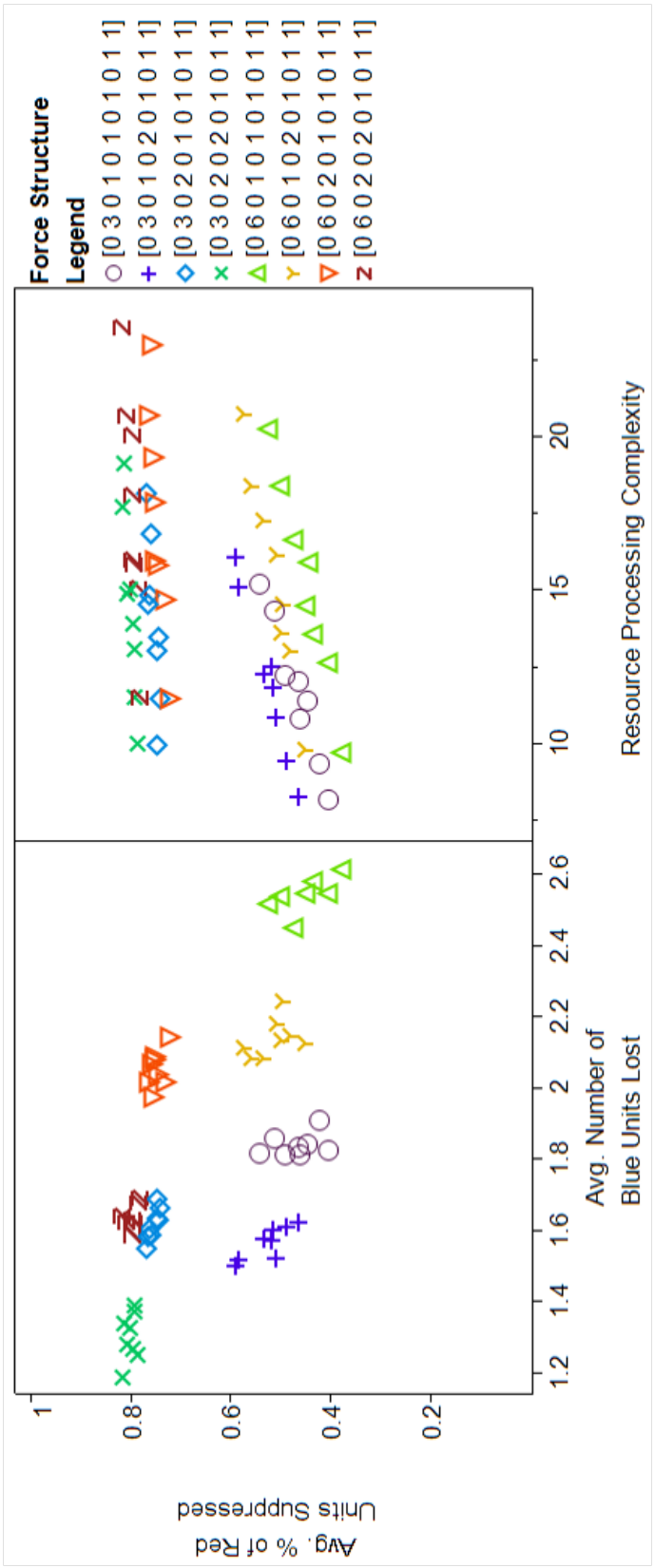


Figure 125: Alternative 11 ARCNET Results

REFERENCES

[1] "Weapon system acquisition reform act of 2009," May 2009.

[2] AbuShaekh, A., Kansal, S., and Levis, A. H., "Modeling Time in DoDAF Compliant Executable Architectures," in *Conference on Systems Engineering Research*, 2007.

[3] AbuSharekh, A., Kansal, S., Zaidi, A., and Levis, A. H., "Modeling Time in DoDAF Compliant Executable Architectures," in *Proceedings of the 2007 Conf. on Systems Engineering Research*, 2007.

[4] Aerospace Systems Design Laboratory, "ARCHITECT: The Architecture-Based Innovation, Technology Evaluation, and Capability Tradeoff Process, ONR Contract Number 000141-01-01-1-2," tech. rep., Georgia Tech Research Corporation, 2011.

[5] Ahmed, S., Wallace, K., Blessing, L., and Moss, M., "Identifying the differences between novice and experienced designers," in *Design for Excellence, the Proceedings of the Engineering Design Conference*, 2000.

[6] Air Force Studies Board, *Pre-Milestone A and Early-Phase Systems Engineering: A Retrospective Review and Benefits for Future Air Force Systems Acquisition.* National Academies Press, 2008.

[7] Alberts, D., Garstka, J., and Stein, F., *Network Centric Warfare*. CCRP, 1999.

[8] Alghamdi, A. and Ahmad, I., "Comparative analysis of defense industry frameworks for c4i system," in *Computer Engineering and Applications (ICCEA), 2010 Second International Conference on*, vol. 1, pp. 443 –447, march 2010.

[9] Anderson, T. W., *An Introduction to Multivariate Statistical Analyses*. John Wiley and Sons, 2003.

[10] ANSI/IEEE, *ANSI/IEEE 1471-2000: Recommended Practice for Architectural Description of Software-intensive Systems*, 2007.

[11] Arrow, K., "A Difficulty in the Concept of Social Welfare," *The Journal of Political Economy*, vol. 58, no. 4, pp. 328–346, 1950.

[12] Bachmann, F., Bass, L., Chastek, G., Donohoe, P., and Peruzzi, F., "The Architecture Based Design Method," cmu/sei-2000-tr-001, Carnige Melon Software Engineering Institute, 2000.

[13] Bagdatli, B., Griendling, K., and Mavris, D., "A Method for Examining the Impact of Interoperability on Mission Performance in a System-of Systems.," in *Proceedings of the 2010 IEEE Aerospace Conference*, 2010.

[14] Balestrini-Robinson, S., *A Modeling Process to Understand Complex System Architectures*. PhD thesis, Georgia Institute of Technology, 2009.

[15] Banks, Jerry, e. a., *Discrete Event System Simulation*. Upper Saddle River, NJ: Pearson Education, Inc, 2010.

[16] Bell, D. and Raiffa, H., *Decision making: Descriptive, normative, and prescriptive interactions*. Cambridge University Press, 1988.

[17] Bienvenu, M. P., Shin, I., and Levis, A. H., "C4ISR architectures: III. An object-oriented approach for architecture design," *Systems Engineering*, vol. 3, no. 4, pp. 288–312, 2000.

[18] Black, D., "On the rationale of group decision-making," *Journal of Political Economy*, vol. 56, no. 1, pp. pp. 23–34, 1948.

[19] Boardman, J. and Sauser, B., "System of Systems - the meaning of of," in *2006 IEEE/SMC International Conference on System of Systems Engineering*, p. 6 pp, April 2006.

[20] Boehm, G. W., "A Spiral Model for Software Development," *IEEE Computer Journal*, vol. 21, pp. 61–72, May 1988.

[21] Bolkcom, C., "Military Suppression of Enemy Air Defenses (SEAD): Assessing Future Needs," tech. rep., CRS Report for Congress, 2005.

[22] Bonabeau, E., "Agent-based modeling: Methods and techniques for simulating human systems," *Proceedings of the National Academies of Science*, vol. 90, pp. 7280–7287, 2002.

[23] Bonacich, P., "Power and Centrality: A Family of Measures," *The American Journal of Sociology*, vol. 92, pp. 1170–1182, 1987.

[24] Bray, J. H. and Maxwell, S. E., "Multivariate Analysis of Variance," *Sage Publications: Quantitative Applications in the Social Sciences*, vol. 54, 1985.

[25] Breyfogle, F. W., *Implementing Six Sigma, Second Edition: Smarter Solutions Using Statistical Methods*. Productivity Press, 2003.

[26] Browning, T., "Applying the Design Structure Matrix to System Decomposition and Integration problems: A Review and New Directions," *IEEE Transactions on Engineering Management*, vol. 48, no. 3, pp. 292–306, 2001.

[27] Busenitz, L. and Barney, J., "Differences between entrepreneurs and managers in large organizations: Biases and heuristics in strategic decision-making," *Journal of Business Venturing*, vol. 12, no. 1, pp. 9 – 30, 1997.

[28] C4ISR AWG, "Levels of Information Systems Interoperability," tech. rep., C4ISR Architecture Working Group co-chaired by J6 and ASD(C3I)/CISALISI, 1998.

[29] Cares, J., *Distributed Networked Operations: The Foundation of Network Centric Warfare*. Newport, RI: Alidade Press, 2005.

[30] Caudle, S., "Homeland Security Capabilities-Based Planning: Lessons from the Defense Community," tech. rep., Naval Postgraduate School ,Center for Homeland Defense and Security, 2005.

[31] Chairman of the Joint Chiefs of Staff, *Chairman of the Joint Chiefs of Staff Instruction CJCSI 3170.01 G*, March 2009.

[32] Charles, P., "Capabilites Based Acquisition... from theory to reality," *CHIPS Magazine*, vol. Summer, pp. 35–39, 2004.

[33] Chen, P. and Clothier, J., "Advancing systems engineering for systems-of-systems challenges," *Systems Engineering*, vol. 6, no. 3, pp. 170–183, 2003.

[34] Chi, Sung-Do, e. a., "FAMES: Fully Agent-Based Modeling and Emergent Simulation.," in *Proceedings of the 2009 Spring Simulation Multi-Conference*, 2009.

[35] Chief Information Officer Council, "A Practical Guide to Federal Enterprise Architecture P," tech. rep., Chief Information Officer, 2001.

[36] Committee on the Planetary Science Decadal Survey; National Research Council, "Vision and Voyages for Planetary Science in the Decade 2013-2022," tech. rep., National Academy of Sciences, 2011.

[37] Dagli, C. and Kilicay-Ergin, N., *System of Systems Engineering: Principles for the 21st Century*, ch. Chapter 4: System of Systems Architecting, pp. 77–100. Wiley, 2009.

[38] Dahmann, J., Lane, J., and Lowry, R., "System Engineering Artifacts for SoS," in *Proceedings of the 5th Annual IEEE System of Systems Enginering Conference*, 2010.

[39] Dahmann, J., Rebovich, G., Lane, J., Lowry, R., and Baldwin, K., "An Implementer's View of Systems Engineering for Systems of Systems," in *Proceedings of the 2011 IEEE International Systems Conference (SysCon)*, no. 978-1-4244-9494-1, (Montreal, QC), pp. 212 – 217, April 4-7 2011.

[40] Defense Acquisition University, *Defense Acquisition Guidebook*, August 5 2010.

[41] Defense Business Board, *Task Group on Capability Requirements Identification and Development Processes Review Briefing.*, October 2008.

[42] Department of Defense, "DoD Modeling and Simulation (M&S) Glossary," 1998.

[43] Department of Defense, *Department of Defense Architecture Framework, version 1.0, Volume I*, August 2003.

[44] Department of Defense, "DoDAF Promulgation Memorandum." Memorandum, February 9 2004.

[45] Department of Defense, *Department of Defense Architecture Framework, version 1.5, Volume I*, April 2007.

[46] Department of Defense, *Department of Defense Architecture Framework, version 1.5, Volume II*, April 2007.

[47] Department of Defense, *Department of Defense Directive 5000.01*, 5000.01 ed., November 20 2007.

[48] Department of Defense, *Joint Publication 3-01: Countering Air and Missile Threats*, 2007.

[49] Department of Defense, *Department of Defense instruction 5000.02*, 5000.02 ed., December 8 2008.

[50] Department of Defense, *Department of Defense Architecture Framework, version 2.0 Volume I*. http://cio-nii.defense.gov/sites/dodaf20/index.html, May 2009.

[51] Department of Defense, *Department of Defense Architecture Framework, version 2.0 Volume II*. http://cio-nii.defense.gov/sites/dodaf20/index.html, May 2009.

[52] Department of Defense, "DoDAF Promulgation Memorandum." Memorandum, May 28 2009.

[53] Dickerson, C. E., Soules, S. M., Sabins, M. R., and Charles, P. H., *Using Architectures for Research Development and Acquisition*. Office of the Assistant Secretary of Defense, 2004.

[54] Dickerson, C. and Mavris, D., *Architecture and Principles of Systems Engineering*. Boca Raton: Auderbach Publications, 2010.

[55] Dickerson, C. and Mavris, D. N., "Relational-Oriented Systems Engineering (ROSE)," in *Proceedings of teh 6th Annual IEEE SoSE Conference, 2011.*, 2011.

[56] Dijkstra, E. W., "T.H. Report 70-WSK-30: Notes on Structured Programming," April 1970.

[57] Director, Systems and Software Engineering, Deputy Under Secretary of Defense (Acquisition and Technology), and Office of the Under Secretary of Defense (Acquisition, Technology and Logistics), *Systems Engineering Guide for Systems of Systems*, version 1.0 ed., August 2008.

[58] Doerry, N. and Fireman, H., "Fleet Capabilities Based Assessment (CBA). White Paper.." White Paper.

[59] Domercant, J. C., *ARC-VM: An Architecture Real Options Complexity-based Valuation Methodology for Military System-of-Systems Acquisition*. PhD thesis, Georgia Institute of Technology, 2011.

[60] Duhaime, I. M. and Schwenk, C. R., "Conjectures on cognitive simplification in acquisition and divestment decision making," *The Academy of Management Review*, vol. 10, no. 2, pp. pp. 287–295, 1985.

[61] Dylla, N., "Experimental Investigation of the Design Process," in *Proceedings of the International Conference on Design Engineering*, vol. 1, 1989.

[62] ECTACO, "ECTACO Electronic Translators," January 2010.

[63] Engler, W., *Architecture and Principles of Systems Engineering*, ch. Chapter 24: Long-Range Strike System Design Case Study, pp. 417–440. Auerbach Publications, 2009.

[64] Estefan, J. A., "Survey of Model-Based Systems Engineering (MBSE) Methodologies." INCOSE, 2008.

[65] Evans, J. H., "Basic Design Concepts," *A.S.N.E. Journal*, pp. 671–678, 1959.

[66] Forsberg, K., Mooz, H., and Cotterman, H., *Visualizing Project Management: Models and Frameworks for Managing Complex Systems*. John Wiley and Sons, 2005.

[67] Forsburg, K. and Mooz, H., "The Relationship of System Engineeirng to the Project Life Cycle," in *Joint Conference of National Council On Systems Engineering (NCOSE) and American Society for Engineering Management (ASEM)*, Center for Systems Management, 1991.

[68] Gansler, J. and Binnendijk, H., eds., *Information Assurance: Trends in Vulnerabilities, Threats, and Technologies*. National Defense University, 2004.

[69] Garcia, J., "Executable architecture analysis modeling," in *Proceedings of the 2007 spring simulation multiconference - Volume 3*, SpringSim '07, (San Diego, CA, USA), pp. 177–184, Society for Computer Simulation International, 2007.

[70] Garcia, J., "Executable Architecture Analysis Modeling for Network Testing and Evaluation in an HLA and TENA environment." White Paper, 2007.

[71] Garcia, J., Browning, J., Paredes-von Oeyen, E., Gupta, A., Manginipalli, C., Bedre, S., and Vepakomma, V., "Innovations in Process Modeling as Applied to JFCOM Joint Experimentation Directorate Division Experiment Management Teams," in *Spring SIW 06S-SIW-015*, 2006.

[72] Garcia, J. and Tolk, A., "Adding executable context to executable architectures: shifting towards a knowledge-based validation paradigm for system-of-systems architectures," in *2010 Summer Simulation Multiconference*, SummerSim '10, (San Diego, CA, USA), pp. 593–600, Society for Computer Simulation International, 2010.

[73] Garlan, D. and Shaw, M., *Advances in Software Engineering and Knowledge Engineering, Volume 1*, ch. An Introduction to Software Architecture. World Scientific Publishing Company, 1993. book author: Ambriola, V; Tortora, G.

[74] Griendling, K., *Architecture and Principles of Systems Engineering*, ch. Chapter 11: DoDAF and MoDAF Artifacts, pp. 157–184. Auderbach Publications, 2010.

[75] Griendling, K. and Mavris, D., "An Architecture-based Approach to Identifying System-of-Systems Alternatives," in *Proceedings of the 5th International Conference on Systems of Systems Engineering*, 2010.

[76] Griendling, K. and Mavris, D., "Development of a DoDAF-based Executable Architecting Approach to Analyze System-of-Systems Alternatives," in *Proceedings of the 2011 IEEE Aerospace Conference*, 2011.

[77] Griendling, K. and Balestrini, S., "DoDAF-based System Architecture Selection Using a Comprehensive Modeling Process and Multi-Criteria Decision Making.," in *proceedings of the 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2008.

[78] Grinstead, C. M. and Shell, L. J., "Introduction to Probability," tech. rep., American Mathematical Society, 2006.

[79] Haas, P. and Shedler, G., "Stochastic Petri Net Representation of Discrete Event Simulations," *IEEE Transactions on Software Engineering*, vol. 15, pp. 381–393, 1989.

[80] Hammond, J., Keeney, R., and Raffia, H., "The hidden traps in decision making," *Harvard Busines*, vol. September-October, 1998.

[81] Hauser, J. R. and Clausing, D., "The House of Quality," *The Harvard Business Review*, vol. May-June, pp. 63–73, 1998.

[82] Headquarters, Department of the Army, *The Army Universal Task List*, August 2003.

[83] Hespeslagh, P. and Jemison, D., *Managing Acquisitions: Creating Value Through Corporate Renewal*. New York: The Free Press, 1991.

[84] Hobday, M., "Product Complexity, Innovation and Industrial Organisation," *Research Policy*, vol. 26, pp. 689–710, 1998.

[85] Hogg, R. and Tanis, E., *Probability and Statistical Inference*. Pearson Prentice Hall, 2006.

[86] Iacobucci, J., *RAAM Rapid Architecture Alternative Modeling: A Framework for Capability-based Analysis of System of Systems*. PhD thesis, Georgia Institute of Technology, To be published May, 2011.

[87] Iacobucci, J. and Mavris, D., "A method for the generation and evaluation of architecture alternatives on the cloud," in *System of Systems Engineering (SoSE), 2011 6th International Conference on*, pp. 137 –142, june 2011.

[88] IEEE Computer Society/Software & Systems Engineering Standards Committee, *IEEE Standard for Application and Management of the Systems Engineering Process (1220-2005)*. IEEE Computer Society/Software & Systems Engineering Standards Committee, March 20 2005.

[89] INCOSE, *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*, version 3.0 ed., June 2006. INCOSE-TP-2003-002-03.

[90] INCOSE Measurement Working Group, "Systems Engineering Measurement Primer," 1998.

[91] INCOSE Measurement Working Group, "Systems Engineering Measurement Primer," 2010.

[92] ISO/IEC/IEEE, *ISO/IEC/IEEE 16326: Systems and software engineering Life cycle processes Project management*, first edition ed., December 15 2009.

[93] ISO/IEC/IEEE, *ISO/IEC FCD 42010 Architecture description*, d8 ed., August 2010.

[94] Jackson, M., *Systems Methodology for the Management Sciences*. Plenum Press, 1991.

[95] Jain, S. and Krishna, S., "Emergence and Growth of Complex Networks in Adaptive Systems.," *Computer Physics Communications*, vol. 121-122, pp. 116–121, 1999.

[96] JCS J-8/Force Application Assessment Division, "White Paper on Conducting a Capabilities-Based Assessment (CBA) Under the Joint Capabilities Integration and Development System (JCIDS)." White Paper, January 2006.

[97] Jemison, D. and Sitkin, S., "Corporate Acquisitions: A Process Perspective," *Academy of Management Review*, vol. 11, pp. 145–163, 1986.

[98] Jensen, M. and Ruback, R., "The Market for Corporate Control: The Scientific Evidence," *Journal of Finacial Economics*, vol. 11, pp. 5–50, 1983.

[99] Johnson, P., *Architecture and Principles of Systems Engineering*, ch. Chapter 23: Air Warfare Model, pp. 397–417. Auerbach Publications, 2010.

[100] Joint Chiefs of Staff, *Joint Publication 3-0 (JP 3-0) Joint Operations*, change 2 ed., September 17 2006.

[101] Joint Chiefs of Staff, *Capabilities-Based Assessment (CBA) User's Guide Version 3*. J-8, March 2009.

[102] Joint Chiefs of Staff, "Joint Publication 1-02 (JP 1-02): DOD Dictionary of Military and Associated Terms," 2009.

[103] Joint Staff, *Universal Joint Task List (UJTL)*. Washington, DC, cjcsm 3500.04d ed., August 2005.

[104] Kadane, J. and Wolfson, L. J., "Experiences in elicitation," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 47, no. 1, pp. 3–19, 1998.

[105] Kaufman, A., *Curbing Innovation: How Command Technology Limits Network Centric Warfare*. Argos Press, 2004.

[106] Keating, C., Rogers, R., Unal, R., Dryer, D., and et al., "System of Systems Engineering," *Engineering Management Journal*, vol. 15, no. 3, pp. 36–45, 2003.

[107] Kidney, W., ed., *Webster's New Compact Dictionary*. Thomas Nelson Inc, 1985.

[108] Kielman, J., Thomas, J., and May, R., "Foundations and Frontiers in Visual Analytics," *Information Visualization*, vol. 8, no. 4, pp. 239–246, 2009. 53199; 239-246; Thomas, Jim; May, Richard; IFMV; 1862743421; 10.1057/ivs.2009.25; Winter 2009; 237205511; 48594671; English; Thousand Oaks; Kielman, Joe; Palgrave Macmillan 2009; PGRVIFMVivs200925.

[109] Kirkwood, C. W., "System Dynamics Methods: A Quick Introduction." Arizona State University College of Business, 2008.

[110] Lawler, G., *Introduction to Stochastic Processes*. Chapman & Hall, 1995.

[111] Levis, A. H., "A Colored Petri Net Model of Intellegent Nodes," in *IMACS Symposium on Modeling and Control of Technological Systems*, 1991.

[112] Levis, A. H. and Wagenhals, L. W., "C4ISR architectures: I. Developing a process for C4ISR architecture design," *Systems Engineering*, vol. 3, no. 4, pp. 225–247, 2000.

[113] Lubatkin, M., "Mergers and the Performance of the Acquiring Firm," *Academy of Management Review*, vol. 8, pp. 218–225, 1983.

[114] Luehrman, T. A., "Investment opportunities as real options: Getting started on the numbers," *Harvard Business Review*, vol. July-August, pp. 4–15, 1989.

[115] Maier, M., "Architecting Principles for System of Systems," *Systems Engineering*, vol. 1, pp. 267–284, 1998.

[116] Mathieson, G., "Best Practice for using Assessment Hierarchies in Operational Analysis Principles and Practical Experiences," in *Proceedings of the 2000 Command and Control Research and Technology Symposium*, 2000.

[117] Matloff, N., "Introduction to Discrete Event Simulation and the SymPy Language," 2008.

[118] Mavris, D. and Griendling, K., "Relational Oriented Systems Engineering and Technology Tradeoff Analysis (ROSETTA) Environment," in *The Proceedings of the 6th Annual IEEE System of Systems Engineering Conference*, 2011.

[119] Mazur, G. H., "The Application of Quality Function Deployment (QFD) to Design a Course in Total Quality Management (TQM) at the University of Michigan College of Engineering," in *Proceedings of International Conference on Quality-1996*, 1996.

[120] Melnyk, C. R., "Petri Nets: An Alternative to Markov Chains," *The Journals of the Reliability Information Analysis Center*, vol. 1st Quarter, 2004.

[121] Merderer, C., "Evolution of Information in a Design Team." Diplom Aribet, University of Munich, 1997.

[122] Meyers, R. H., Montgomery, D. C., and Anderson-Cook, C. M., *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. John Wiley & Sons, 2009.

[123] Meyn, S. P., *Control Techniques for Complex Networks*. Cambridge University Press, 2007.

[124] Mittal, S., "Extending DoDAF to Allow DEVS-based Modeling and Simulation," *Special issue: DoDAF, Journal of Defense Modeling and Simulation JDMS*, vol. III, no. 2, 2006.

[125] Mooz, H. and Forsberg, K., "A Visual Explanation of Development Methods and Strategies including the Waterfall, Spiral, Vee, Vee+, and V++ Models." Center for Systems Management, Inc., 2001.

[126] Morkevivius, A., gudas, S., and Sinlingas, D., "Model-Driven Quantitative Performance Analysis of UPDM-based Enterprise Architecture," 2010.

[127] NASA, "NASA Systems Engineering Processes and Requirements w/Change 1 (11/04/09)."

[128] NASA, *Software Assurance Standard*, nasa-std-8739.8 with change 1 ed., July 28 2004.

[129] NASA, *NASA Systems Engineeing Handbook*. NASA, NASA Headquarters, Washington, D.C. 20546, sp-2007-6105 rev 1 ed., December 2007.

[130] National Defense Industrial Association, "National Defense Industrial Association: Systems Engineering," January 2010.

[131] National Defense Industrial Association Systems Engineering Division Task Group, *Top Five Systems Engineering Issues in Defense*, version 8 ed., August 18 2006.

[132] Office of Management and Budget, "Circular A-130: Management of Federal Information Resources," 2000.

[133] Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering, "Defense acquisition program support methodology," January 2009.

[134] Olhanger, J. and West, B. M., "The House of Flexibility: Using the QFD Approach to Deploy Manufacturing Flexibility," *International Journal of Operations and Production Management*, vol. 22, pp. 50–79, 2002.

[135] Olmez, A. E., "Mission Centric Reliability Analysis of C4ISR Architectures Using Petri Nets," in *Proceedings of the IEEE SMC Conference*, 2003.

[136] OMG, "The Unified Profile for the Department of Defense Architecture Framework (DoDAF) and the Ministry of Defence Architecture Framework (MODAF)," 2009.

[137] O'Rourke, R., "Navy Network-Centric Warfare Concept: Key Programs and Issues for Congress," tech. rep., Foreign Affairs, Defense, and Trade Division, 2004.

[138] Oxford English Dictionary, "Oxford English Dictionary Online: Architecture," November 2009.

[139] Pablo, A. L., Sitkin, S. B., and Jemison, D. B., "Acquisition decision-making processes: The central role of risk," *Journal of Management*, vol. 22, no. 5, pp. 723 – 746, 1996.

[140] Panconesi, A., "The Stationary Distribution of a Markov Chain," tech. rep., La Sapienza University of Rome, 2005.

[141] Pankin, M. D., "Baseball as a Markov Chain.," January 10 2010.

[142] Parnas, D. L., "On the Criteria to be Used in Decomposing Systems into Modules," *Commissions of the Association for Computing Machinery*, vol. 15, pp. 1053–1058, 1972.

[143] Pawlowski, T., "Executable Architecture Methodology for Analysis." Presentation, 2004.

[144] Pawlowski, T., Barr, P., and Ring, S., "Applying Executable Architectures to Support Dynamic Analysis of C2 Systems," in *2004 Command and Control Research and Technology Symposium: The Power of Information Age Concepts and Technologies*, 2004.

[145] Perry, W. e. a., "Measures of Effectiveness for the Information-Age Navy: The Effects of Network-Centric Operations on Combat Outcomes," tech. rep., RAND, 2002.

[146] Princeton University Word Net, "Princeton University Word Net version 3.0," December 2009.

[147] ReVelle, J. B., Moran, J. W., and Cox, C. A., *The QFD Handbook*. John Wiley & Sons, Inc, 1998.

[148] Ritchey, T.and Zwicky, F., "Morphologie and Policy Analysis," in *16th EURO Conference on Operational Analysis*, 1998.

[149] Ritchey, T., *Wicked Problems/Social Messes: Decision Support Modelling with Morphological Analysis*. Springer, 2011.

[150] Ross, D., "Structured Analysis (SA): a language for communicating ideas," in *IEEE Transactions on Software Engineering SE-3(1)*, p. 1634, 1977.

[151] Royce, W. W., "Managing the Development of Large Software Systems," in *IEEE WESCON Proceedings*, pp. 328–338, IEEE, 1970.

[152] Rumsfeld, D., "Requirements System Memorandum to Gen. Pace." Memorandum, March 18 2002.

[153] Sage, A. P. and Cuppan, C. D., "On the Systems Engineering and Management of Systems of Systems and Federations of Systems," *Inf. Knowl. Syst. Manag.*, vol. 2, no. 4, pp. 325–345, 2001.

[154] Schwenk, C. R., *Identity, Learning, and Decision Making in Changing Organizations*. Praeger, 2011.

[155] Schwenk, C. R., "Cognitive Simplification Processes in Strategic Decision-Making," *Strategic Management Journal*, vol. 5, no. 2, pp. pp. 111–128, 1984.

[156] Scott, M. J., *Handbook for Conducting Gap Analysis*. Department of Fish and Wildlife, University of Idaho, 2007.

[157] Singh, H. and Montgomery, C. A., "Corporate acquisition strategies and economic performance," *Strategic Management Journal*, vol. 8, no. 4, pp. pp. 377–386, 1987.

[158] System Dynamics Society, "What is System Dynamics," November 2010.

[159] System of Systems Engineering Center of Excellence, sponsored by the Office of the Under Secretary of Defense for Acquisition, Technology, & Logistics, Defense Systems, Systems and Mission Integration, Joint Force Integration (USD-AT&L), "System of Systems Engineering Center of Excellence: SoS Engineering," August 2010.

[160] Tako, A. A. and Robinson, S., "Model building in System Dynamics and Discrete-event Simulation: A Quantitaive Comparison," in *System Dynamics Society*, 2008.

[161] Taylor, R. A. ., "Origin of System Dynamics: Jay W. Forrester and the History of System Dynamics." US Department of Energy's Introduction to System Dynamics, 2008.

[162] Thomas, J. and Cook, K., eds., *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. National Vizualization and Analytics Center, 2005.

[163] TRADOC Army Capabilities Integration Center, "Capabilities-Based Assessment (CBA) Guide v. 2.0.," May 12 2009.

[164] Ullman, D., "The Ideal Engineering Design Support System," tech. rep., Robust Decsions, Inc. written for Autodesk, 2000.

[165] Ullman, D., *The Mechanical Design Process*. McGraw Hill, 2009.

[166] United States Air Force, *Air Force Task List (AFTL)*, August 1998.

[167] United States Air Force, *Early Systems Engineering Guidebook*, version 1 ed., 2009.

[168] United States Congress, "Clinger Cohen Act of 1996." Information Technology Management Report Act, 1996.

[169] United States Department of Defense, "Joint Vision 2020," tech. rep., 2000.

[170] United States Department of Defense, "Network Centric Warfare: Department of DefenseReport to Congress," tech. rep., 2001.

[171] United States Department of the Navy, *Naval Systems Engineering Guide*, October 2004.

[172] United States General Accounting Office, "INFORMATION TECHNOLOGY MANAGEMENT: SBA Needs to Establish Policies and Procedures for Key IT Processes," Tech. Rep. GAO/AIMD-00-170, Report to the Chairman, Committee on Small Business, U.S. Senate, 2000.

[173] United States Government Accountability Office, "DEFENSE ACQUISITIONS: DODs Requirements Determination Process Has Not Been Effective in Prioritizing Joint Capabilities." GAO-08-1060, 2008.

[174] United States Government Accountability Office, "DEFENSE ACQUISITIONS: Fundamental Changes Are Needed to Improve Weapon Program Outcomes." GAO-08-1159T, 2008.

[175] United States Government Accountability Office, "DEFENSE ACQUISITIONS: DOD Must Prioritize Its Weapon System Acquisitions and BalanceThem with Available Resources." GAO-09-501T, March 25 2009.

[176] United States Navy, *Universal Naval Task List (UNTL)*, November 2005.

[177] U.S. Marine Corps, *SUPPRESSION OF ENEMY AIR DEFENSES (SEAD)*, mcwp 3-22.2 ed.

[178] Volovoi, V., "Modeling of System Reliability Using Petri Nets with Aging Tokens.," *Reliability Engineering and System Safety*, vol. 84, pp. 149–161, 2003.

[179] Volovoi, V., "Stochastic Petri Modeling Using SPN@." White Paper, 2006.

[180] Wagenhals, L. W., Shin, I., Kim, D., and Levis, A. H., "C4ISR architectures: II. A structured analysis approach for architecture design," *Systems Engineering*, vol. 3, no. 4, pp. 248–287, 2000.

[181] Wells, G. and Sage, A., *System of Systems Engineering: Principles for the 21st Century*, ch. Chapter 3: Engineering of a System of Systems, pp. 44–76. Wiley, 2009.

[182] West, D., *Introduction to Graph Theory*. Prentice Hall, 2007.

[183] Yassine, A., "An Introduction to Modeling and Analyzing Complex Product Development Processes Using the Design Structure Matrix (DSM) Method," *Quaderni di Management (Italian Management Review)*, vol. 9, 2009.

[184] Zwicky, F., *Discovery, Invention, Research Through the Morphological Approach*. New York: Macmillan, 1st american edition ed., 1969.